AFIT/GE/EE/80D-45

AUTOMATIC RECOGNITION OF PHONEMES USING A SYNTACTIC PROCESSOR FOR ERROR CORRECTION

THESIS

AFIT/GE/EE/80D-45 Robert B. Taylor
2Lt USAF

Approved for public release; distribution unlimited.

AFIT/GE/EE/80D-45

AUTOMATIC RECOGNITION OF PHONEMES USING A SYNTACTIC PROCESSOR FOR ERROR CORRECTION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

by

Robert B. Taylor, B.S.E.E.
2Lt USAF
Graduate Electrical Engineering

December 1980

Preface

This work has been motivated by my interest in artificial intelligence and pattern recognition which was originally sparked by the work of Dr. Bruce D. Fritchman, Professor of Electrical Engineering at Lehigh University and which was reinforced by a course in pattern recognition taught by Dr. Matthew Kabrisky, Professor of Electrical Engineering at the Air Force Institute of Technology.

I gratefully acknowledge the advice, the encouragement, and the stimulating environment provided by Dr. Kabrisky as my thesis advisor, and by Capt. Larry R. Kizer and Professor Charles W. Richard as members of my thesis committee. In addition, I wish to express my appreciation to Lt. Paul Dundas and Lt. Mark Felkey for their comments concerning my work and for their pleasant company.

Finally, I owe my greatest appreciation to my wife, Mary Anne, and to my two children, Stephanie and Sybil. For without their unabated love and support during this difficult period, I could not have completed this thesis project.

Robert B. Taylor

Contents

List of Figures

AFIT/GE/EE/80D-45

Abstract

An algorithm was developed and partially implemented to integrate the use of a phoneme recognizer and a syntactic error corrector for continuous speech recognition. The recognizer uses LPC reflection coefficients as a feature set and makes decisions based on the computation of pairwise likelihood ratio tests for M phonemes. The syntactic error corrector uses a backtracking parser to perform phonological rule and grammatical error correction. A computer program is included to provide interactive training with a Tektronix 4010 terminal on a Data General NOVA/ECLIPSE computer system.

AUTOMATIC RECOGNITION OF PHONEMES USING A SYNTACTIC PROCESSOR FOR ERROR CORRECTION

I. Introduction

With the recent advances in computer technology, emerging applications for that technology, and the need to communicate more rapidly and efficiently, much attention is being focused on the use of an automatic speech recognizer (ASR) to fill one gap in the man/machine interface. These applications include very low bit rate transmission of speech, automatic translation of speech into written text, voice controlled cockpits, etc. In each of these applications, the human operator should be relatively free to perform several tasks simultaneously yet more efficiently than ever before.

With only a little bit of imagination, then, one could dream up dozens of other applications for speech recognition. However, the question at hand is not how we would be able to apply speech recognition systems, but, rather, how we would implement them, and further, how much we are willing to pay to attain a desired level of recognition accuracy.

The pitfalls in the method of implementation are numerous. By way of introduction, one problem area is the resolution of ambiguities that can occur in everyday spoken language. The phrases "fire a round" and "fire around" are spoken exactly the same in many dialects of English. However, their meanings are totally different. To the human listener, this would pose no problem since the phrases could be distinguished by the sentence context. But to the processor in a voice controlled cockpit, the phrase might be interpreted as "open fire on enemy target" or, "fire into the vicinity of .." or, alternatively, "there is a fire around .." depending on what other information was available. The cost of responding incorrectly to an utterance could be extremely high in this case. In another environment, say an application in voice transcription, an ambiguity would also be present although the cost of an error may or may not be as high depending on the sensitivity of the text.

The purpose of this example was to illustrate that in some applications, it is not only desireable, it is imperative that the ASR have the ability to resolve ambiguities to some degree. Some details of implementing this capability will be discussed later.

The task of automatic speech recognition entails extracting from a continuously varying acoustic speech signal a set of features which contains the information essential to the recognition process. A simplification of this procedure and the comparison to human speech recognition is shown in Figure 1. These features can then be mapped into a set of discrete symbols representing linguistic units. Although these units could be words or syllables, the most frequently used unit is the phoneme.

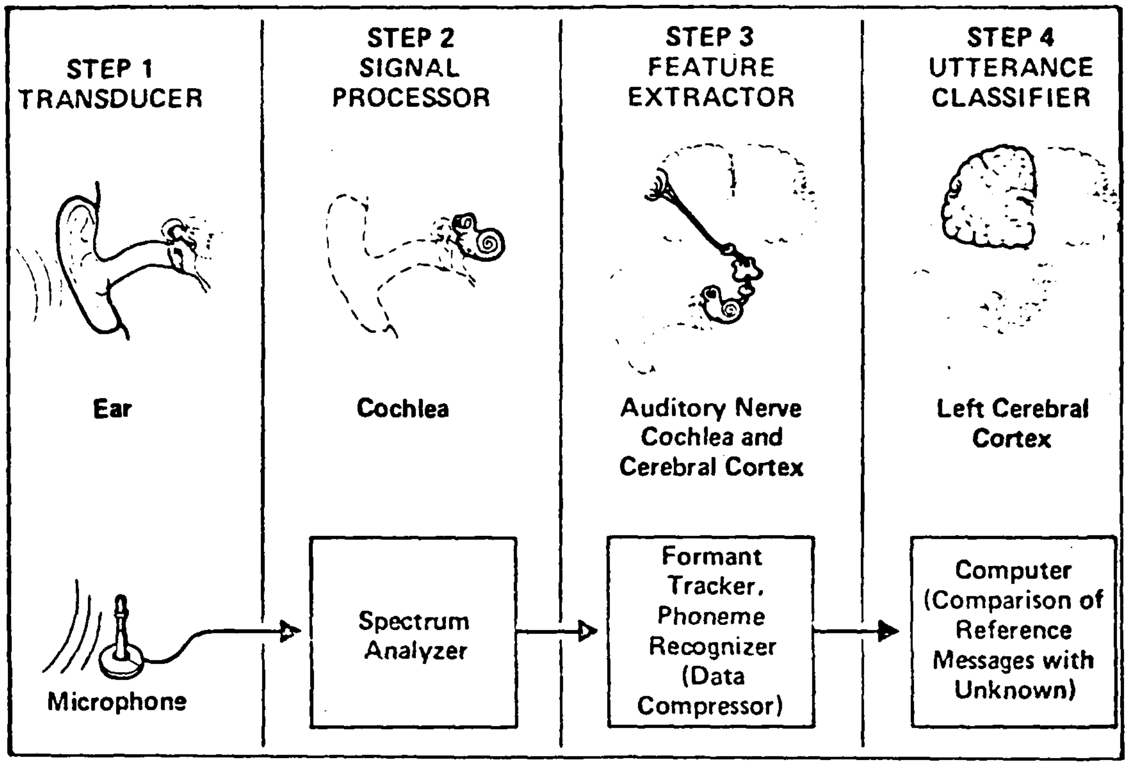

Figure 1. Comparison of Human to Machine Recognition (Ref. 18)

The representation of speech by phonemic strings is not altogether exact, however, since linguistics is merely a model of the speech process. In fact, the degree of variability in the production and interpretation of speech is virtually infinite. For example, consider an experiment in which normal conversational speech is recorded and displayed on a speech spectrogram. The spectrogram is then independently translated by several linguists. To illustrate this, the hand labelling of the utterance, "Edit the phonemic labels," is shown in Figure 2.

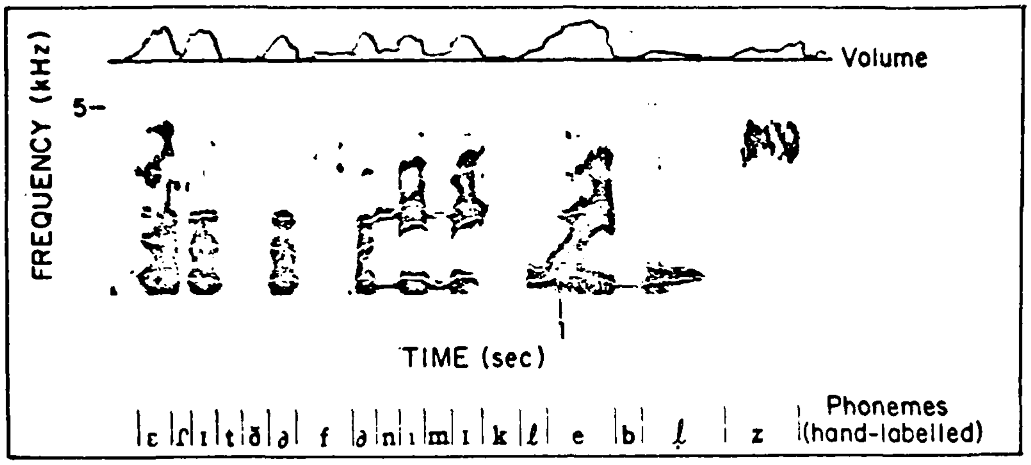

Figure 2. Hand Labelling a Speech Segment (Ref. 17)

Almost certainly, there would be some differences between the respective linguists' interpretations. There would also probably be differences between their translations and the accepted standard phonetic transcription found in a pronunciation dictionary.

Fortunately, the redundancy in natural languages allows the listener to understand the message most of the time. This is because he is usually able to reconstruct a meaningfully correct sentence from the recognized portions of the speech through his understanding of the phonetic, syntactic, and semantic rules of the language.

Therefore, a useful ASR must also incorporate knowledge of these rules to ensure adequate performance of the recognizer. Unfortunately, this often adds almost unsurmountable complexity to an already difficult problem.

The general speech recognition problem can be separated into several sub-tasks with varying levels of difficulty. Most of these sub-tasks are treated separately in the literature. The first level of difficulty is with the recognition of an isolated word. Further complexity is added if this is to be done by more than one speaker or if the word comes from a large vocabulary.

The second level of difficulty is that encountered in word spotting from continuous speech. Again, added complexity comes about with the above requirements of speaker independence and large vocabulary size.

The most difficult level of speech recognition, the one addressed in this paper, is with the application to speech understanding or continuous speech transcription.

Problem

The objective of this research was to develop an algorithm to be used in a continuous speech recognizing system (CSR) that would unify current efforts in feature extraction and syntactic error correction. This research is necessary because, for the most part, developments in these two areas are being made almost totally independently of one another. Consequently, when these two technologies have been integrated in a single system, the familiar "black box" approach was typically used.

To cite an example of the potential power of syntactic error correctors, Woods (Ref. 21:355) pointed out that, because of the inevitability of decision errors with forced decision rules ("hard" decisions), nondeterministic algorithms (utilizing "soft" decision rules) have a tremendous advantage for CSR's. These nondeterministic algorithms systematically consider all possible sequences of linguistic units (phonemes, words, etc.) until one of the sequences yields a successful analysis. Success might depend on various criteria such as syntactic or semantic validity of the utterance.

A nondeterministic algorithm is only one form of a syntactic error corrector. However, to ensure that many of the existing syntactic error correctors will be usable with the end result of this research, several criteria were placed on the CSR algorithm to be developed:

1. It must provide <u>at least</u> one phoneme as an alternative for each observation interval.

2. It must have a potential for real-time implementation.

3. It must afford ease of adaptation to changes in technology and changes in empirical knowledge about the speech process.

The last criterion was imposed to allow flexibility of recognition performance since there is still much to learn concerning what tradeoffs exist when combining phoneme decision rules with syntactic error correctors and semantic knowledge.

Scope

The scope of this research includes the development of an algorithm for a "soft" phoneme decision rule to be used in a CSR. This decision rule is to meet the three major criteria stated above.

In addition to the development of the phoneme decision rule, a simple CSR system will be described which can be used to test the validity of the assumptions made and the validity of the theoretical results of this research. This implementation is intended to be an experimental implementation of the decision rule and not a final system design.

Assumptions

It is assumed here that any errors existing in the measurement of the acoustic speech signal are caused by an additive white gaussian noise process. It is also assumed that the noise process is stationary and ergodic. The effects of quantization errors in the A/D sampling of the speech are ignored. While these assumptions add to the tractibility of the mathematics in the theory and are commonly made assumptions, it is not presently known whether they are actually valid here or if they accurately reflect the physics of the CSR problem.

Approach and Presentation

The presentation of this research effort is, for the most part, in the order that the work was done. In Chapter II, a detailed analysis of the CSR problem is given and some of the inadequacies of existing strategies are stated.

In Chapter III, a well known model of the speech process is discussed. From this model, parameters are selected as the feature set for a simple CSR system. Then, Chapter IV explains the control strategy for an experimental CSR system. These two chapters are presented before the following two chapters to introduce terminology and serve as further motivation for the material presented in Chapters V and VI.

Chapter V introduces some background in statistical decision theory that is necessary for the development of the theory in Chapter VI. It also contains results for binary-hypothesis decision rules that will be used in the next chapter.

Chapter VI contains the major contribution of this research. It develops an M-ary decision rule for non-mutually exclusive decision regions. A solution algorithm is also described to find the thresholds for the pairwise likelihood ratio tests (LRT) that make up the decision rule.

Chapters VII and VIII conclude the paper with a summary of the results and recommendations for future work. Following the main body of the paper is an appendix containing computer programs which implement the algorithms discussed in the paper.

II. Detailed Analysis

As mentioned before, automatic speech recognition with application to speech understanding systems or automatic speech transcription is considerably more difficult than word spotting or isolated word recognition. It is a problem characterized by huge amounts of data, high data rates, and a great deal of uncertainty about the data source. Moreover, the dynamics of speech such as phoneme duraticn, volume, pitch, and rate of pronunciation are extremely difficult to deal with in a single speech model.

However, before discussing the model of speech used in this paper, it is appropriate to first discuss some of the considerations that motivated the choice of the speech model and the overall design.

Although varying with the application, the most important consideration is to attain high recognition accuracy. In general, speech recognition for military applications will require very high recognition accuracies and very efficient algorithms. These two requirements appear to be conflicting and perhaps they are at the present. But, this condition is relative to the state of technology at any given time. For this reason, it might be logical to approach the problem by first fixing an upper limit to the "cost" that is tolerable. Under this constraint, we could then seek to maximize the performance of the Continuous Speech Recognizer (CSR).

For the general problem of maximizing some objective function subject to fixed constraints, there exist several solution techniques. One solution technique that nicely fits the problem at hand is the Lagrange multiplier technique (Ref. 2:18-21). As will be seen later, use of this method can be made to find the optimum thresholds of a likelihood ratio test (LRT) for a particular performance desired. It is this strategy that is adopted in this paper since we can reassign the desired performance to bring the costs within tolerable limits.

Because of the inevitability of making decision errors when decisions are forced, it is desirable to retain alternative phonemes for later consideration. This idea of "deferred decision" has its foundations in the area of sequential decision theory. Most applications of this theory are associated with radar detection problems. However, it will be shown that we can make use of the basic idea of deferring a phoneme decision with an associated loss due to the deferral. In the radar context, this loss might be associated with the cost of going back to a particular sector in a search volume to ascertain whether a target is or is not present before alerting a tracking radar. In our problem the cost could be associated with the overhead of retaining more than one possible phoneme for a particular speech segment.

Obviously, if we retained all possible phonemes for each speech segment, we would later be able to generate all possible combinations of phonemes that could have been uttered. But, this procedure would tell us nothing we couldn't have deduced before the words were even spoken. In addition, the cost in memory requirements for retaining alternative phonemes would be exceeded only by the cost in time it would take to generate all possible sentences made up by the phonemes.

To overcome the problems of forced decision rules in isolated word recognition systems, several authors such as Itakura (Ref. 5) and Kashap (Ref. 6) have recently reported on algorithms which minimize the overall "distance" between a hypothesized string and the observed string of phonemes. This procedure (dynamic programming), although improving the word recognition performance remarkably for isolated word applications, has serious problems when applied to continuous speech recognition. The primary difficulty here is determining the boundaries between words in the observed utterance.

The difficulty with word boundaries can be understood by thinking about how a sentence is typically spoken. It is usually one continuous utterance with no break between words. In fact, when there are breaks in the acoustic waveform, it is most frequently caused by stop sounds such as "p", "d", "t", etc. rather than by gaps between adjacent words. Therefore, it can easily be seen that word boundaries would cause considerable difficulties in continuous word speech.

If a minimum distance rule of the variety used in isolated word recognition were applied on continuous word recognition, there would probably be disastrous results. For example, if two words of different lengths exist in the vocabulary with similar initial pronunciation, the wrong word could very easily have the least "distance". Then, if the minimum distance rule were applied, the estimate of the word boundary would be thrown off. This would then give rise to false recognition of other words in the following speech segments until "resynchronization" with the actual word boundaries was accomplished.

An alternative to the forced (or "hard") decision rule, as was mentioned earlier, is the deferred (or "soft") decision rule. To implement this strategy, however, it must be determined at what point to make the cutoff between retaining additional phonemes. Retaining the additional phonemes constitutes a requirement to eliminate the incorrect ones at some later time.

To accomplish this correction, we can make use of the results in the field of linguistics. Each of us has innate knowledge of the phonological rules governing the production of speech. That is, our knowledge of English tells us that certain strings of phonemes are permissable and others are not. For example, when a word begins with the sound "l" or "r" we know that a vowel should follow. Anything other than a vowel violates the restrictions we know exist.

Even more importantly, our knowledge of the rules of phonology allows us to tolerate the variability in the production of sounds between different dialects and even for a single speaker utterring a sound in a different context. One such rule tells us that voiceless stop sounds such as "p" are aspirated at the beginning of a word. This means that we should observe a more breathy quality to the production of the phoneme than, for example, if it occurred in the middle of the word. This rule and others are typically formalized in a quasi-equation form called the Backus-Naur Form (BNF). Most of the rules we know that are applicable in the English language can be found tabulated in this form in (Ref. 1). By making use of this knowledge, we could automatically eliminate some of the phonemes retained by the deferral process.

In addition to this knowledge, linguistics gives us rules which allows us to understand sentences that we may or may not have ever heard before. This is because of our knowledge of the allowable structures of sentences. For example, the sentence, "The nine month old baby took me to work," is perfectly understandable although it doesn't make much sense.

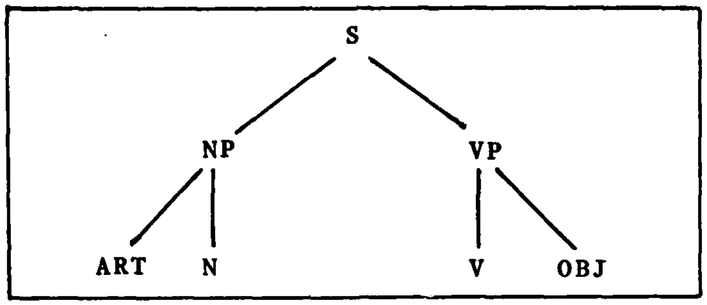

Figure 3. Sample Grammar

We are able to understand sentences like the above because of relatively few grammatical rules that allow us to construct grammatically correct sentences from even a small vocabulary, or lexicon. When you consider all the subjects, adjectives, verbs, and objects that can be formed from the huge lexicon that most people possess, an enormous number of sentences can be formed using the simple grammar shown in the tree diagram in Figure 3.

It should be obvious that although not all the possible sentences conforming to such a grammar would have a logical semantic meaning, the use of this kind of syntactic knowledge can eliminate the majority of the erroneous sentences allowed by the deferred decision process.

The final primary consideration discussed here is that of efficiency. Because most applications for speech recognition would require real-time or close to real-time speed, the algorithms implemented must be simple enough to be implemented in hardware. It is also desireable that the algorithms be easily adaptible to new technologies. That is, since technology is constantly evolving but not every aspect evolves as rapidly as another, it would be most cost effective to implement the algorithms in modules. That way, when it appeared that technology in one area affected one but not all of the modules, it could still be incorporated in a revision of the design with a minimum of effort.

A good example of this is the increasing likelihood that Charge Coupled Devices (CCD) will be playing an increasingly important role in signal processing applications such as spectral estimation. As these special purpose devices become more and more available, the implementation of algorithms could be changed. However, the performance wouldn't be changed unless the algorithms were also changed. The big impact of these devices, then, would be in the tolerable cost. That is, if speed of computation were increased, it would be possible to increase the performance by simply adjusting the thresholds of the decision rule.

III. The Speech Model

The purpose of this chapter is to discuss the necessary elements of a speech model for the proposed recognition system. Because the actual speech model used is not the primary concern of this paper, we will concentrate on the Linear Predictive Coding (LPC) speech model.

Before discussing the LPC model in detail, however, a brief account will be given on the development of speech models.

History of Speech Models

The earliest models of speech production were mechanical and date back as far as the late 1700's (Ref. 3:166). Typical of these mechanical models were the use of bellows to force air past a reed which, in turn, excited hand variable resonators.

Later mechanical models, such as shown in Figure 4, were more complicated but were also more versatile. In the reed chamber, tension on the reed was variable to control the pitch while the control keys changed the resonances corresponding to the various shapes the vocal tract takes on during speech.

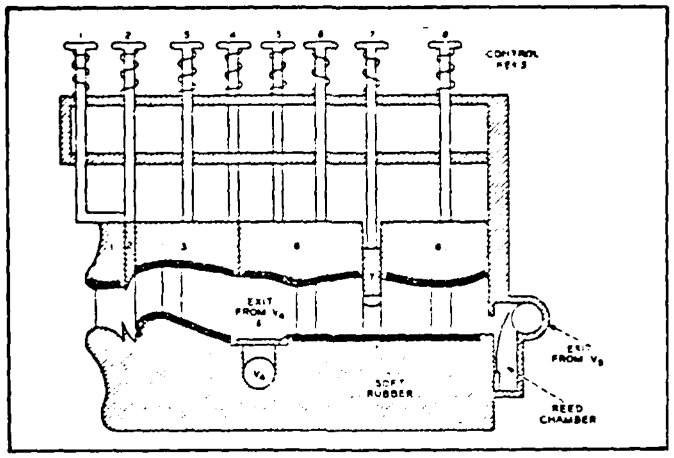

Figure 4. Mechanical Vocal Tract (Ref. 3)

All of the early mechanical models were attempts at simulating the dynamic movements of the glottis (vocal chords), the tongue and lips. Obviously, the vocal tract shown in Figure 5A is a complex mechanism to model acurately.

More recently, attempts were made to model the vocal tract mathematically. One of these models, called the acoustic tube model, utilized the theory of fluid motion through a series of adjacent cylinders (Figure 5B). Reflection coefficients have been derived for this model (Ref. 7:61-71) which uniquely characterize the tube for any possible combination of diameters, or areas, of the individual cylinders (Figure 5C).

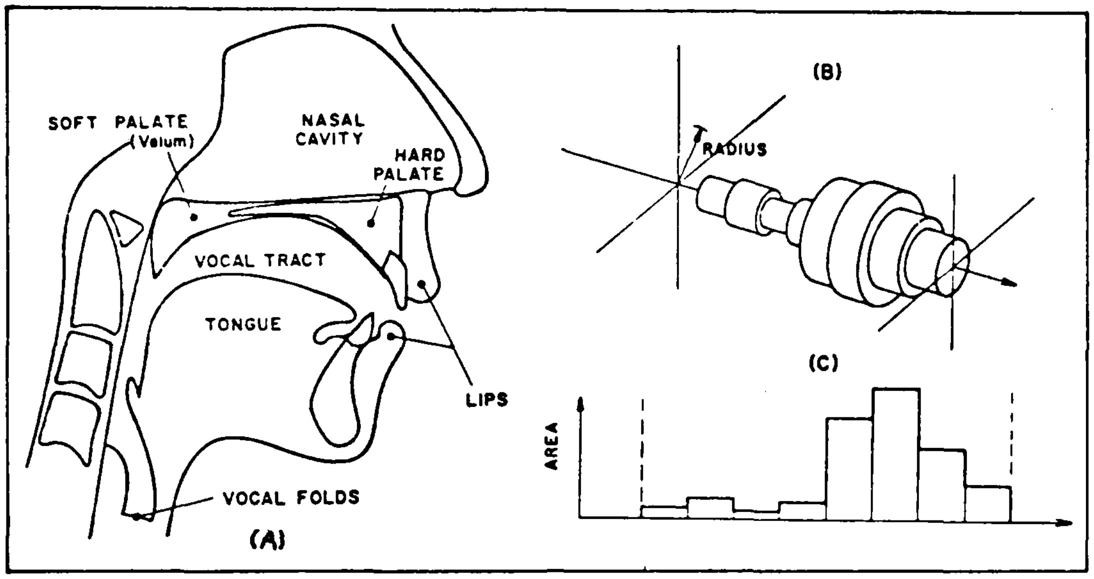

Figure 5. Vocal Tracts: A) Human B) Acoustic Tube C) Area Function (Ref. 7)

At about the same time that work was being done on the acoustic tube model (ca. 1970), the LPC speech model was also evolving. This model captured the attention of many researchers currently working on the speech problem. To date, LPC is the most widely used and most versatile tool for the analysis of speech.

LPC in Speech Analysis

The LPC model is, essentially, a stochastic finite difference equation with constant coefficients. The a(i)'s in equation (3-1) are assumed constant, but only over short time intervals. The sequence e(n) represents the residual prediction error and is assumed to be samples of a white noise process during unvoiced speech and an impulse train during voiced speech (Ref. 5:67-72). X(n) is the result of a prediction based on the previous P samples of speech.

$$x(n) = e(n) + \sum_{i=1}^{P} a(i)x(n-i) \qquad (3-1)$$

LPC has rapidly become a well known tool for the analysis and transmission of speech. Its popularity is due primarily to its success as a data compression technique. It is well known that speech can be represented much more compactly if, instead of using the original speech samples or the Fourier coefficients of a speech segment, the values of the LPC predictor coefficients are used. To illustrate the dramatic reduction in data possible, variable bit rate schemes in speech transmission systems can achieve a data rate as low as 1,200 bits/second compared to the 40,000 to 200,000 bits/second rate required to transmit the raw sampled-data speech (Ref. 7:246).

Another application of the LPC technique is in power spectrum estimation in the speech signal. Although also a consideration for the previous application, the question of non-stationarity of the speech signal becomes more apparent for this case since the spectrum is clearly non-stationary. Thus, the spectrum that can be computed from the LPC predictor coefficients is only an estimate of the true spectrum. Therefore, in order to reduce the variance in the estimate of the short time power spectrum, we are forced to do spectral averaging or windowing of the speech samples (Ref. 12:532-570). In fact, if we did spectral averaging over all time, we would be assured of having an asymptotically consistent estimate with a variance equal to that obtainable for a stationary process. This would, however, reduce the time resolution of the power spectrum that is desirable for speech spectrograms. Hence, there is a tradeoff between the uncertainty of the spectral estimate and the time resolution. Several windowing techniques (e.g. Hanning, Hamming, and Kaiser-Bessel) that are commonly used to attain this tradeoff are discussed in (Ref. 12:239) and (Ref. 13:88).

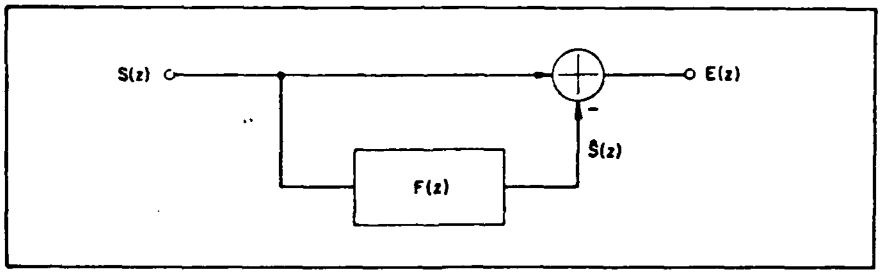

Figure 6. Linear Predictor Block Diagram (Ref. 7)

A simplification of the LPC algorithm is shown in Figure 6. S(z) is the Z-Transform of the discrete time, sampled speech. F(z) represents the transfer function of the predictor filter which has, as its input, S(z). The output of the filter is $\hat{S}(z)$, the maximum likelihood estimate of S(z).

The actual implementation of the LPC algorithm for speech analysis is not typically carried out in digital filters as is suggested by Figure 6. Rather, the sampled speech data is normally the input to algorithms such as Levinson's recursion (Ref. 7:55) running on large scale computers which perform the analysis offline.

Parametric Representations of LPC

The term parametric representation of speech is in reference to the characterization of the speech waveform by a reduced set of parameters. It could be thought of as the mapping of a point from a multi-dimensional coordinate system to another coordinate system of lower dimension. The advantage of the mapping rests in the fewer number of coordinates required to specify a unique point in the new coordinate system.

The five parametric representations discussed here all have a common factor. That is, each of the other four parameter sets can be derived from the original predictor coefficients. These predictor coefficients are normally the basic outputs of the LPC algorithm.

Unfortunately, it is well known that the predictor coefficients are very sensitive to finite word length effects inherent in small computers which frequently results in instability in the model (Ref. 7:229).

One set of parameters directly transformable from the predictor coefficients is the set of autocorrelation coefficients. Theoretically, stability of the digital filter is guaranteed due to the properties of the positive definite Toeplitz matrix encountered in the solution procedure (Ref. 19:29). However, errors in calculations, again attributed to finite computer word length, can destroy the positive definiteness of the matrix resulting in instability.

The spectrum coefficients can also be calculated from the predictor coefficients by solving for the roots of the predictor filter F(z) (Ref. 7:229). These roots are called the spectrum coefficients because they determine the amplitude spectrum of the digital filter in the Z-Transform domain. By solving for these parameters and linearly interpolating between successive iterations, stability can be guaranteed when using as few as five bits per parameter (Ref. 7:229).

Another transformation of the predictor coefficients which assures stability is the set known as the cepstrum (Ref. 7:229-230). This set, however, has the disadvantage of requiring many operations including a logarithmic transformation on the magnitude spectrum. Obviously, this kind of computational burden is not desirable for a real-time implementation.

Although several other transformations have been discussed in the literature, only one other set of parameters has properties worthy of mention for this application. This set, the PARCOR coefficients, or reflection coefficients, has been shown to be equivalent to the reflection coefficients used to describe the acoustic tube model previously mentioned (Ref. 16:417). Since these coefficients may also be solved for recursively (Ref. 7:55), they are very attractive for implementation on a small computer.

One additional property of the reflection coefficients is that the transformation yielding them has been shown to orthogonalize the basis vectors in an appropriate Hilbert space (Ref. 9:33). This result implies that for the same number of parameters, this set will be less sensitive to the noise present in the speech signal than any other set.

IV. Control Strategy

Now that a set of parameters has been selected as the feature set, namely the LPC reflection coefficients, a simple system that makes use of these features can be presented. A block diagram of the continuous speech recognizer (CSR) is given in Figure 7.

The first component of the CSR is the signal conditioner. It performs the analog to digital (A/D) conversion of the continuous speech waveform and does a first order pre-emphasis to accent the higher formants. These formants, or resonances of the vocal tract, are difficult to discern visually without pre-emphasis. For this system, pre-emphasis is implemented with a digital filter of the form 1 - aZ where a = 1.0 is used. This value yields a pre-emphasis of approximately 6dB / octave (Ref. 7:166).

The pre-emphasized LPC model spectrum for the utterance, "which way did you walk," is shown in Figure 8. Notice the upward-going trajectory of the 2nd formant in the "AEE" sound of "way" which is located between the 41st and 49th time segments.

The final part of the signal conditioner is the LPC algorithm itself. N samples from a disk file containing the A/D speech are input to the LPC algorithm for each execution of the algorithm.

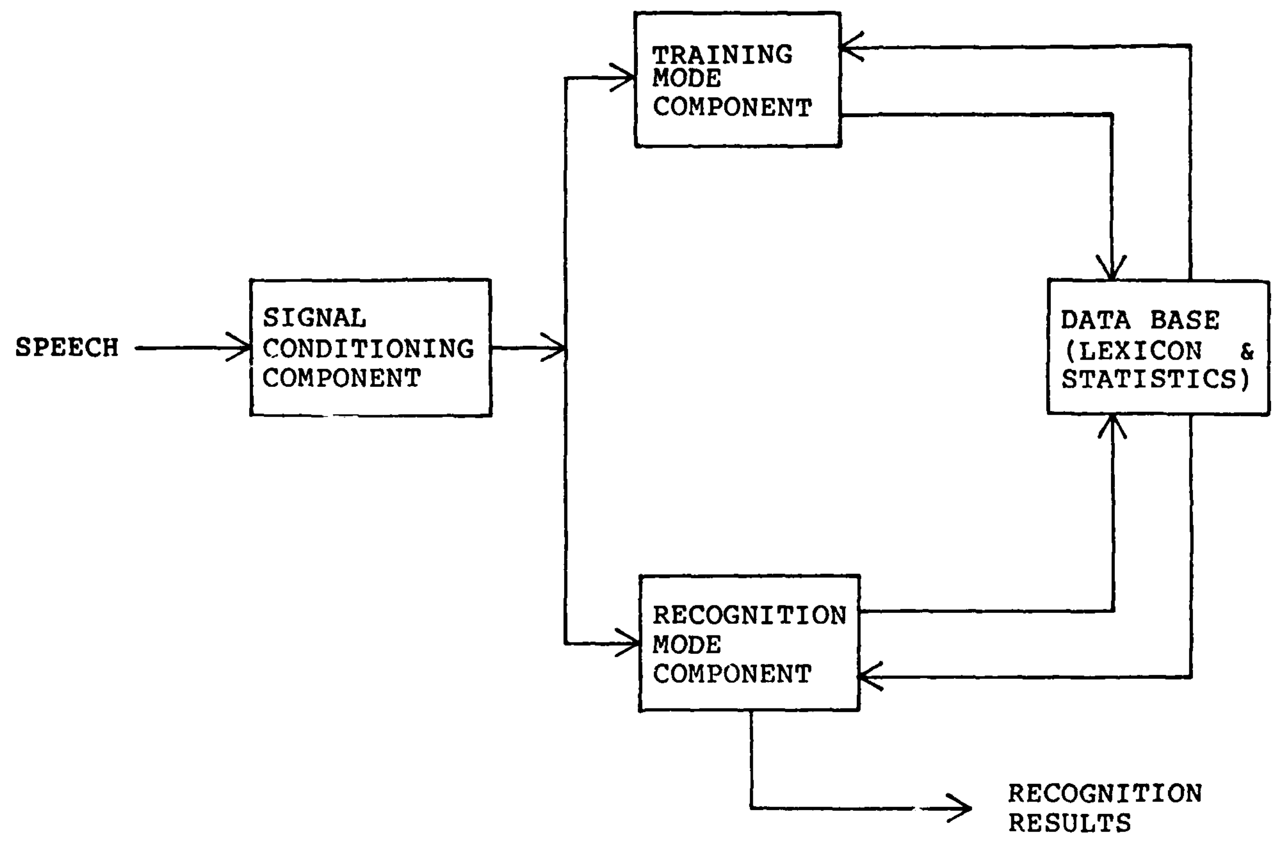
SPEECH
SIGNAL CONDITIONING COMPONENT
TRAINING MODE COMPONENT
RECOGNITION MODE COMPONENT
DATA BASE (LEXICON & STATISTICS)
RECOGNITION RESULTS

Figure 7. CSR Components

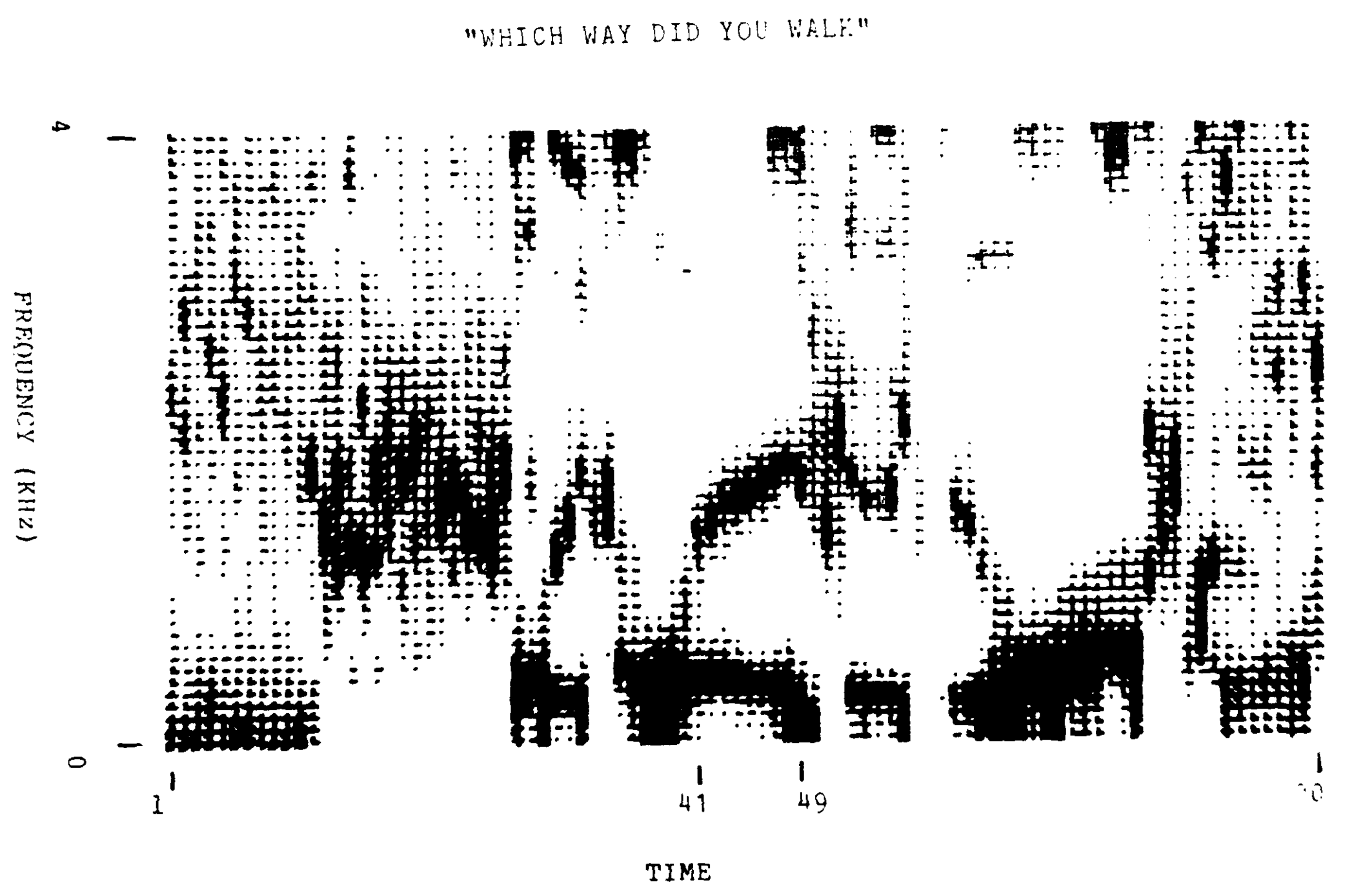

Figure 8. LPC Model Spectrum

These N samples constitute a speech segment, or frame, and typically represent several milliseconds of speech. For an 8KHz sample rate, N=64 to N=256 is appropriate.

From these N samples, a feature vector containing P LPC parameters is calculated. It is this vector which is used in both the training mode and the recognition mode to classify speech.

The second component of the CSR is the training mode. A flow diagram for this component is given in Figure 9. This component is necessary to generate the decision thresholds of a likelihood ratio test (LRT) to be used later in the recognition mode. Generation of the thresholds is detailed in Chapter VI.

The training mode can be entered as often as is necessary to increase recognition accuracy. Initially, it will be necessary to enter this mode to build a set of decision thresholds for a single speaker. Conducting training sessions over a period of several days will ensure that the statistics on the LPC features will account for much of the single speaker variability that is known to exist.

The training mode was perceived as a potentially time consuming process. Therefore, since training will likely be performed on an ongoing basis, an interactive program called CLASSIFY (Appendix A) was written to minimize the burden of

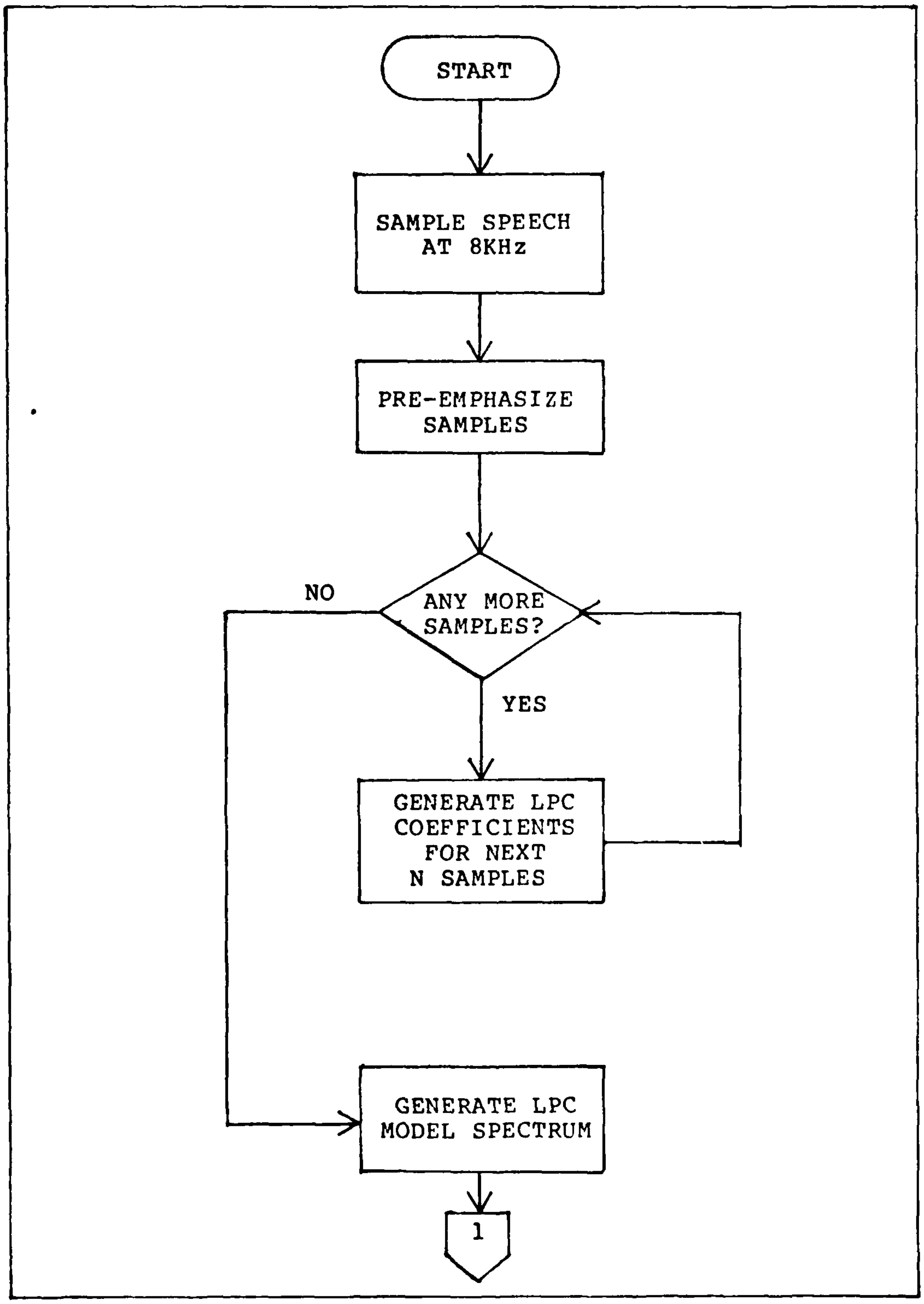
START
SAMPLE SPEECH AT 8KHz
PRE-EMPHASIZE SAMPLES
ANY MORE SAMPLES?
NO
YES
GENERATE LPC COEFFICIENTS FOR NEXT N SAMPLES
GENERATE LPC MODEL SPECTRUM
1

Figure 9. Training Session

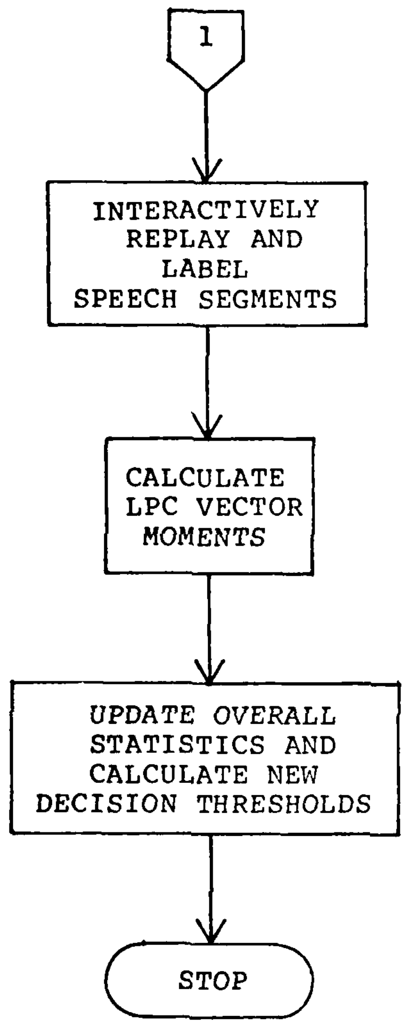
1
INTERACTIVELY REPLAY AND LABEL SPEECH SEGMENTS
CALCULATE LPC VECTOR MOMENTS
UPDATE OVERALL STATISTICS AND CALCULATE NEW DECISION THRESHOLDS
STOP

Figure 9. (Cont.)

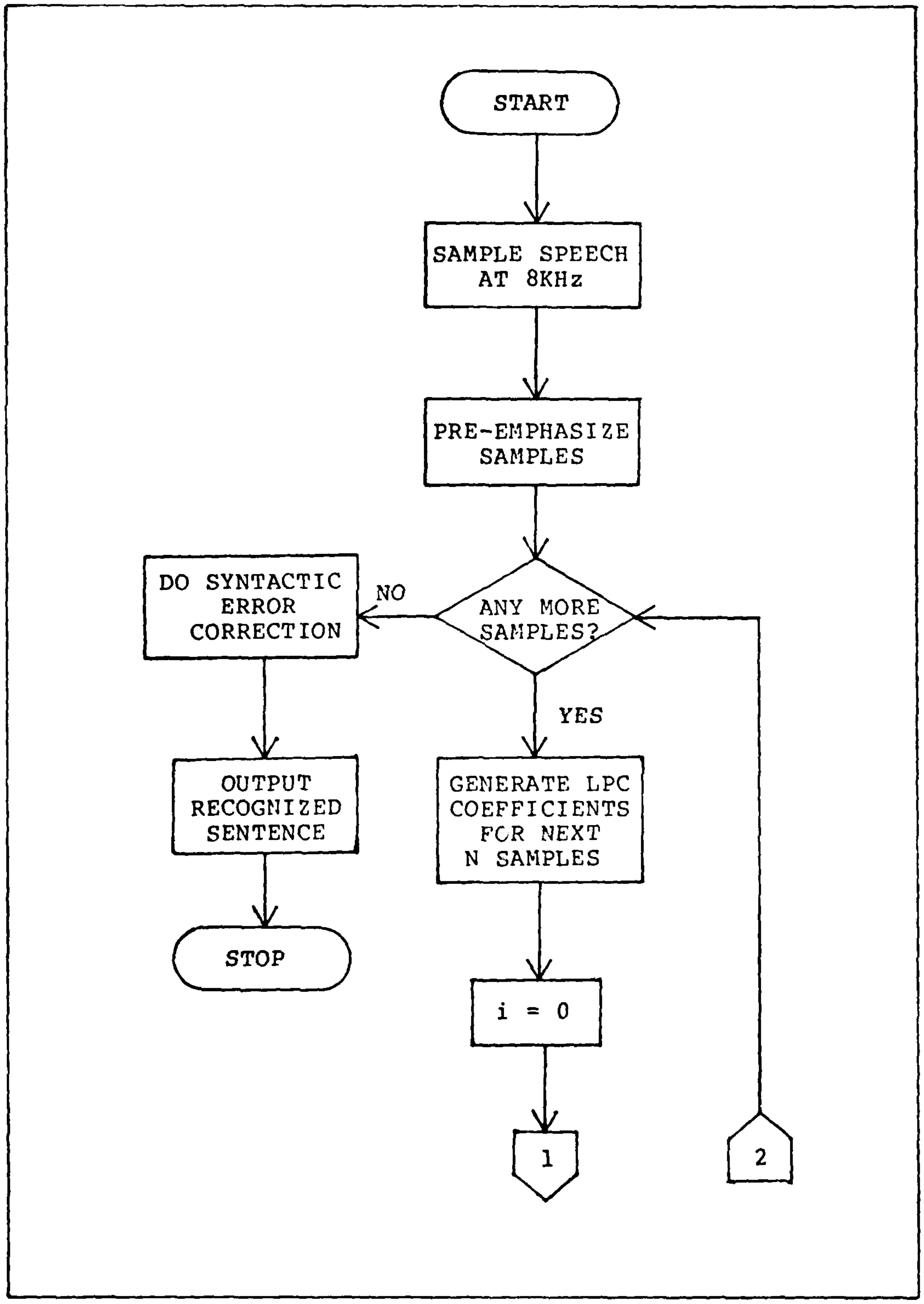
START
SAMPLE SPEECH AT 8KHz
PRE-EMPHASIZE SAMPLES
ANY MORE SAMPLES?
NO
DO SYNTACTIC ERROR CORRECTION
OUTPUT RECOGNIZED SENTENCE
STOP
YES
GENERATE LPC COEFFICIENTS FOR NEXT N SAMPLES
i = 0
1
2

Figure 10. Recognition Session

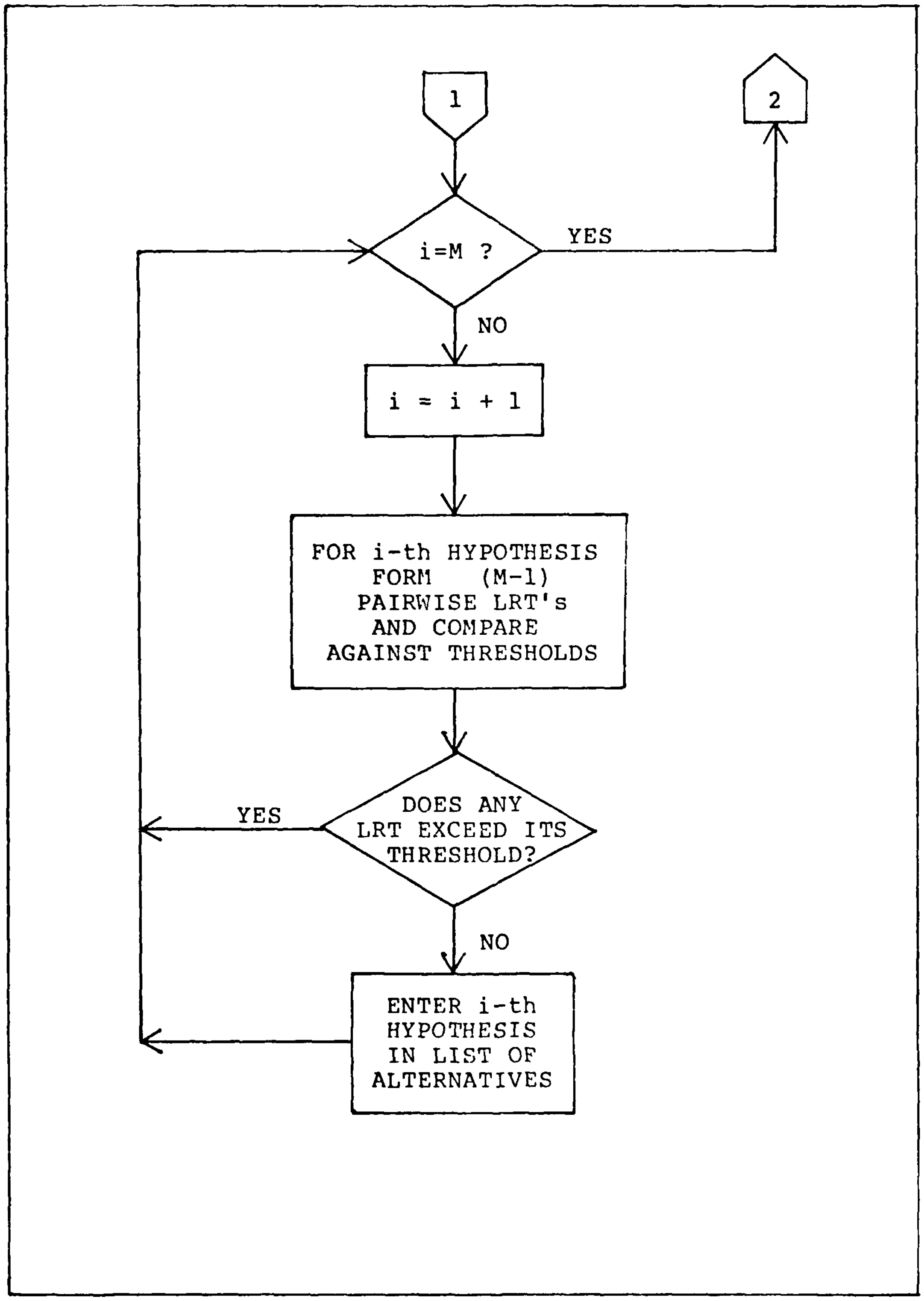
1
2
i=M ?
YES
NO
i = i + 1
FOR i-th HYPOTHESIS FORM (M-1) PAIRWISE LRT's AND COMPARE AGAINST THRESHOLDS
DOES ANY LRT EXCEED ITS THRESHOLD?
YES
NO
ENTER i-th HYPOTHESIS IN LIST OF ALTERNATIVES

Figure 10. (Cont.)

experimentation. With CLASSIFY, the user can view the LPC model spectrum on a Tektronix 4010 terminal. The resultant display is much like that of a typical speech spectrogram. Intensity modulation is achieved by using the grey tone patterns discussed in the text on interactive computer graphics by Newman and Sproull (Ref. 10:225-27).

CLASSIFY uses the interactive interrogative capabilities of the Tektronix terminal by displaying movable crosshairs on the display surface. With these crosshairs, the user can type a two character sequence to label the starting and ending segments of a phoneme. These boundaries must be located visually by the user.

Optionally, the user can get an audible replay of the labelled segments to ensure the validity of his visual recognition. This is accomplished by doing a digital to analog (D/A) conversion of the original speech samples corresponding to the labelled speech segments. At the end of the training session, the classifications are optionally saved and updates of the LPC vector moments are automatically made.

The next component of the CSR is the recognition component shown in Figure 10. As with the training mode, the recognition algorithm begins with sampling and pre-emphasis of the speech. Again too, the LPC coefficients are generated for each speech segment. However, in this mode, each observation vector is used to form M*(M-1) pairwise LRT's since there are (M-1) pairwise LRT's for each of the M hypotheses.

As will be shown in Chapter VI, the i-th phoneme will be considered as one of the alternative choices if and only if all of the M-1 LRT's associated with the i-th hypothesis have values below their respective decision thresholds. With this procedure, there is a potential that, at one extreme, M choices exist, or, at the other extreme, no choices exist. If the former occurs, then no entry will be made in the list of alternatives. For every other case, however, an entry will be made for each hypothesis that satisfies the above test.

A typical time history for the output of the decision rule during a recognition session is shown in Figure 11. Here, each phoneme is represented by a two character machine-readable code. The codes used by this experimental system consists of a subset of the codes used by Cohen and Mercer (Ref. 1). In Figure 11, the first and fourth speech segments have alternative phonemes which are then input to the next phase of the recognizer.

The next two phases of the recognition mode are dependent on the application. For this simple CSR, a backtracking syntactic parser (Ref. 21:355) performs the error correction. This type of parser is very similar to the parsers in simple compilers.

AX	XX	PX	TH	UU	SX	TX	EH	RX
AH			DH					
AW								
UU								
1	2	3	4	5	6	7	8	9

TIME $\longrightarrow$

Figure 11. Phoneme Decision Rule Output

In fact, the actual parser that is implemented here is a modification of a general parser due to Wirth (Ref. 20:304-7) and is given in Appendix A.

The operation of this backtracking parser is very simple but can also be very time consuming. Whenever the input phoneme list consists of more than one alternative, all the other choices are stored on a pushdown stack. The parser then parses like a deterministic one, that is, without backtracking, until either it encounters an invalid sequence or it reaches a satisfactory parse. If the sequence was invalid, the program backs up one step, popping the next phoneme off the top of the stack. This procedure is reiterated until either a successful parse occurs or until all the possible phoneme sequences have been exhausted.

Two important conditions can result with this approach:

1. No valid sequences are found.
2. A valid sequence was detected before all sequences were exhausted.

With the occurrence of the first condition, it was decided to simply print out all possible phoneme sequences in the form of Figure 11. This is done as a preliminary solution to the problem in order to gain insight for future solutions. When the second condition exists, it was decided, again for experimental purposes, to continue the analysis and print out all valid phoneme sequences.

At this point, it should be reiterated that the implementation of the syntactic error corrector was not the major objective of this research. Rather, it was to develop a unified approach to the interfacing of a phoneme decision rule to a syntactic error corrector. In addition, it was required that the performance of the speech recognition be readily adaptible to subjective criteria. With this in mind, the next two chapters deal with the development of such an algorithm. The next chapter provides the background for Chapter VI where a decision rule is actually developed.

V. Statistical Hypothesis Testing

This chapter will discuss some well known techniques for the statistical approach to pattern recognition. The Bayes classifier will be presented for introductory purposes and is shown only for the binary hypothesis case. The Neyman-Pearson criterion is also presented for the binary case and an extension of this is made to the M-ary hypothesis problem in the next chapter.

Bayes Decision Rule for Minimum Error

There are at least three well known strategies for implementing a Bayes classifier (Ref. 15:23-33). The first to be discussed here uses minimum probability of error as a decision criterion.

The objective of this technique is to determine whether a particular observation vector, $\underline{R}$, belongs to hypothesis H_0 or H_1. The decision rule based on the a posteriori probabilities $p(H_i|\underline{R})$ may be written

$$p(H_1|\underline{R}) \underset{H_o}{\overset{H_1}{\gtrless}} p(H_o|\underline{R}) \qquad (5\text{-}1)$$

This is read, if the probability that H_1 occurred, given the observation $\underline{R}$, is greater than the probability that H_0 occurred, tnen we choose H_1; if not, choose H_0.

The test is difficult to use in this form, however, since the a priori probabilities are more readily available than the a posteriori probabilities. An alternate form can be derived using Bayes' rule which says

$$p(H_i|\underline{R}) = \frac{p(\underline{R}|H_i)\, p(H_i)}{p(\underline{R})} \tag{5-2}$$

Since $p(\underline{R})$ is common to both sides of inequality (5-1) the factors will cancel each other when substituting (5-2). Using this fact and writing $p(H_i)$ as P_i, the decision rule of (5-1) can be expressed as

$$L(\underline{R}) \triangleq \frac{p(\underline{R}|H_1)}{p(\underline{R}|H_o)} \underset{H_o}{\overset{H_1}{\gtrless}} \frac{P_o}{P_1} \triangleq \eta \tag{5-3}$$

$L(\underline{R})$ is called the likelihood ratio and should be recognized as the most basic quantity in statistical hypothesis testing. As will be shown, it will also appear as the basic quantity in the Neyman-Pearson criterion.

The use of the LRT with the present decision criterion involves comparing $L(\underline{R})$ to η, the threshold. If $L(\underline{R})$ exceeds the threshold, then we choose H_1; if not, we choose H_0.

An equivalent form of the LRT is the log-likelihood ratio test. That is,

$$\log\left[L(\underline{R})\right] \triangleq \log\left[\frac{p(\underline{R}|H_1)}{p(\underline{R}|H_o)}\right] \underset{H_o}{\overset{H_1}{\gtrless}} \log\eta \tag{5-4}$$

yields identical results as the LRT since the log function is monotonic and both sides of the inequality (5-3) are positive (Ref. 15:27). From this point on we will refer to both the LRT and the log-LRT as simply the LRT.

Although this decision rule will tell us which hypothesis to pick based on the known probabilities, there is still a finite probability that the rule will lead to decision errors. The major drawback to this criterion is that it ignores the cost of making incorrect decisions.

Bayes' Decision Rule for Minimum Risk

The minimum risk criterion addresses the problem that results when we have to pay more of a penalty for making one kind of decision error than we would for making another kind. For example, in a radar scenario, we may pay dearly for failing to detect an enemy target while the occurrence of a false alarm might be much more tolerable.

For the binary hypothesis case there are four costs:

C_{00} = cost of saying H_0 when H_0 was true,
C_{01} = cost of saying H_0 when H_1 was true,
C_{10} = cost of saying H_1 when H_0 was true,
C_{11} = cost of saying H_1 when H_1 was true.

The Bayes risk function involving these costs is

$$\mathcal{R} = C_{oo}P_o \int_{Z_o} p(\underline{R}|H_o)\, d\underline{R} + C_{o1}P_1 \int_{Z_o} p(\underline{R}|H_1)\, d\underline{R}$$

$$+ C_{1o}P_o \int_{Z_1} p(\underline{R}|H_o)\, d\underline{R} + C_{11}P_1 \int_{Z_1} p(\underline{R}|H_1)\, d\underline{R} \qquad (5\text{-}5)$$

where Z is the entire observation space of which $\underline{R}$ could be a member. Z_0 and Z_1 are partitions of Z such that if the observed vector $\underline{R}$ maps into Z_i, we will say H_i. This is illustrated in Figure 12.

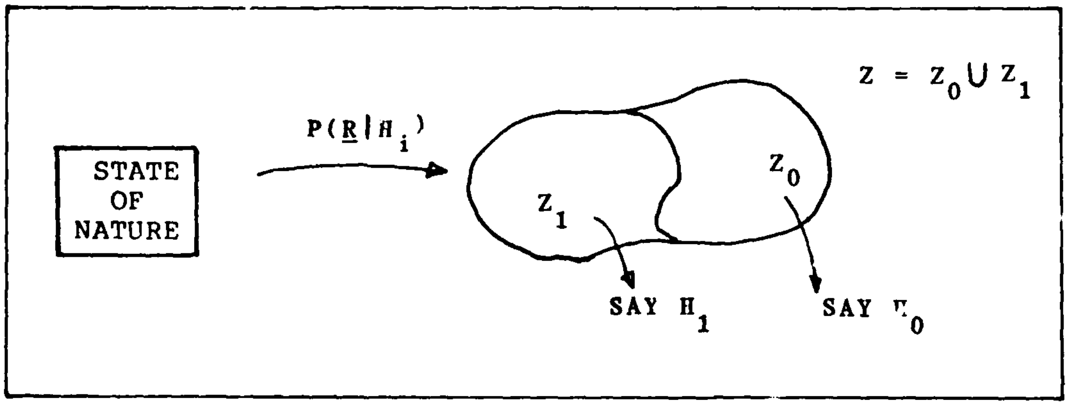

Figure 12. Partitioning the Observation Space

If we are forced to decide for each observation $\underline{R}$, the expression for the Bayes risk in (5-5) suggests that the optimum test is given by partitioning Z into Z_0 and Z_1 such that, for each possible $\underline{R}$, the risk is minimized. This can be achieved by manipulating the expression in (5-5) into an integration only over Z_0 using the fact that $Z_1 = Z - Z_0$. The risk function is then reduced to

$$\mathcal{R} = P_o C_{1o} + P_1 C_{11} + \int_{Z_o} \left\{ \left[P_1 (C_{o1} - C_{11}) p(\underline{R}|H_1) \right] - \left[P_o (C_{1o} - C_{oo}) p(\underline{R}|H_o) \right] \right\} d\underline{R} \qquad (5\text{-}6)$$

But, we know that probabilities are always non-negative and we can assume that the costs are also non-negative. If, in addition, it is assumed that the cost of a wrong decision is greater than the cost of a correct one ($C_{01} > C_{00}$, $C_{01} > C_{11}$, etc.), then the risk will be minimized by assigning $\underline{R}$ to Z_0 if and only if the integrand in (5-6) is negative since the first two terms represent fixed costs. The optimum test, then, can be written in terms of the integrand of (5-6) as

$$P_1 (C_{o1} - C_{11}) p(\underline{R}|H_1) \underset{H_o}{\overset{H_1}{\gtrless}} P_o (C_{1o} - C_{oo}) p(\underline{R}|H_o) \qquad (5\text{-}7)$$

or,

$$L(\underline{R}) \triangleq \frac{p(\underline{R}|H_1)}{p(\underline{R}|H_o)} \underset{H_o}{\overset{H_1}{\gtrless}} \frac{P_o (C_{1o} - C_{oo})}{P_1 (C_{o1} - C_{11})} \triangleq \eta \qquad (5\text{-}8)$$

Thus, the basic quantity in the test is again an LRT.

Mini Max Test

Although it was shown above that the minimum Bayes risk test is optimum when the a priori probabilities P_0 and P_1 and all the costs are known, there are applications in which only the costs are known reliably. In these cases, using the mini-max criterion will assure an upper bound on the risk function.

In words, the mini-max procedure is merely a more generalized version of the minimum Bayes risk criterion. That is, the values of P_0 and P_1 are assumed to take on some value between 0 and 1 and the associated Bayes risk is calculated for the given costs. This procedure is repeated for all combinations of P_0 and P_1. The resulting Bayes risk is then usually plotted against P_1 (fixing P_1 also specifies P_0). Then, to determine the value of the threshold in the LRT, the value of P_1 is assumed to take on the value at which a line tangent to the curve plotted has minimum slope (Ref. 15:31-33) (Ref. 14:67-69).

Neyman-Pearson Criterion

The above criteria for hypothesis testing result in almost identical tests which are computationally simple especially when the probability rules governing the observations, $p(\underline{R}|H_i)$, can be assumed to be gaussian. The question arises, however, about the validity of using these criteria when both the cost factors and the *a priori* probabilities, P_i, are not reliably known.

A different point of view for testing statistical hypotheses was presented in the original paper by Neyman and Pearson which can be found in a volume of collected papers

(Ref. 11:140-185). The arguments given there were, in effect, a re-evaluation of the intent of statistical tests.

The authors reasoned that for every value a continuous observation vector, $\underline{R}$, could take on, there is zero a priori probability (a singularity) that $\underline{R}$ should have taken on the value that it did. The consequence of this is that, by itself, a statistical test cannot provide conclusive evidence for either the truth or falsehood of a hypothesis concerning the state of nature. Rather, if the test were used as a rule upon which we accepted or rejected the hypothesis, then we would be correct more often than not in the long run.

Although this statement is not terribly profound, the authors go on to argue that there is little evidence that a single criterion for deriving the test enjoys universal success. This is true, they state, because in many problems of importance, the cost factors and probabilities are arrived at subjectively or empirically.

It would appear reasonable, then, that more than one criterion may be suitable for testing a particular hypothesis.

Associated with a test on hypothesis H_i, there are two types of errors that can occur. Type I is the error resulting from rejecting H_i when it is true while Type II is the error resulting from accepting H_i when some alternative hypothesis, H_i' is true.

The Neyman-Pearson criterion assumes that there may be several suitable criteria, for example, to minimize the chance of a Type I error. Since any of those criteria would be suitable, then, it would be reasonable to pick one that would also simultaneously minimize the chance of a Type II error. However, since this is usually a conflicting objective, an alternative is to minimize the probability of one type of error subject to a constraint on the other. Neyman and Pearson showed that the solution of this problem is equivalent to the solution of a problem in the Calculus of Variations.

In a radar problem, a common procedure is to maximize the probability of detection, P_D , subject to a constraint on the probability of a false alarm P_{FA}. The Lagrange multiplier method has been shown to be a useful technique in the solution of this problem (Ref. 15:33-34).

First, a loss function, F, is defined in terms of $P_M = 1 - P_D$ and P_{FA}. Specifically,

$$F = P_M + \lambda \left[P_{FA} - \alpha' \right] \tag{5-9}$$

where λ is a Lagrange multiplier ($\lambda \geq 0$) and $\alpha' < \alpha = P_{FA}$. Then, if $P_{FA} = \alpha'$, the second term is zero and in order to maximize P_D, we must minimize F (or P_M).

By definition,

$$P_M \triangleq \int_{Z_0} p(\underline{R}|H_1)\, d\underline{R} \qquad (5\text{-}10)$$

and

$$P_{FA} \triangleq \int_{Z_1} p(\underline{R}|H_0)\, d\underline{R} \qquad (5\text{-}11)$$

Then, F can be written

$$F = \int_{Z_0} p(\underline{R}|H_1)\, d\underline{R} + \lambda \left[\int_{Z_1} p(\underline{R}|H_0)\, d\underline{R} - \alpha' \right] \qquad (5\text{-}12)$$

F can then be written in terms of an integral only over Z_0

$$F = \lambda(1-\alpha') + \int_{Z_0} \left[p(\underline{R}|H_1) - \lambda\, p(\underline{R}|H_0) \right] d\underline{R} \qquad (5\text{-}13)$$

It can now be seen that to minimize F for a <u>fixed</u> value of λ, we simply assign to Z_0 only those points $\underline{R}$ which would result in a negative contribution to the integrand of (5-13). This again results in the LRT

$$L(\underline{R}) = \frac{p(\underline{R}|H_1)}{p(\underline{R}|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \lambda \quad . \qquad (5\text{-}14)$$

However, to satisfy the constraint on P_{FA}, we must choose λ so that the expression for P_{FA} in (5-11) yields $\alpha' = P_{FA}$. Or, equivalently, we must solve for λ in (5-15).

$$P_{FA} = \int_{\lambda}^{\infty} p\left[L(\underline{R})|H_o\right] dL(\underline{R}) = \alpha' \qquad (5\text{-}15)$$

In the case where the joint densities $p(\underline{R}|H_i)$ are gaussian, the pdf can be written as

$$p(\underline{R}|H_i) = \frac{1}{(2\pi)^{\frac{N}{2}} |K_i|^{\frac{1}{2}}} \exp\left| -\frac{1}{2} (\underline{R}-\underline{m}_i)^T Q_i (\underline{R}-\underline{m}_i) \right| \qquad (5\text{-}16)$$

where $\underline{m}_i$ is the mean vector, K_i is the covariance matrix of $\underline{R}$, and Q_i is the inverse of K_i under the ith hypothesis.

In the special case where the N components of $\underline{R}$ are also independent (implies an orthogonal basis), the joint density reduces to a product of densities of the individual components:

$$p(\underline{R}|H_i) = \frac{1}{(2\pi)^{\frac{N}{2}} \prod_{j=1}^{N} \sigma_{ij}} \exp\left| -\frac{1}{2} \sum_{j=1}^{N} \frac{(r_j - m_{ij})^2}{\sigma_{ij}^2} \right| \qquad (5\text{-}17)$$

Now, concentrate on the exponent of the ratio in (5-14) (formed by using (5-17) with i=1 in the numerator and i=0 in the denominator). Calling this ratio d, we can merge terms and complete the square.

$$d = \frac{1}{2} \sum_{j=1}^{N} \frac{(r_j - m_{oj})^2}{\sigma_{oj}^2} - \frac{1}{2} \sum_{j=1}^{N} \frac{(r_j - m_{1j})^2}{\sigma_{1j}^2} \qquad (5\text{-}18)$$

Or,

$$d = \frac{1}{2}\sum_{j=1}^{N} \frac{\sigma_{1j}^2 - \sigma_{oj}^2}{\sigma_{oj}^2\ \sigma_{1j}^2} \left\{ \left[r_j + \frac{(m_{1j}\sigma_{oj}^2 - m_{oj}\sigma_{1j}^2)}{\sigma_{1j}^2 - \sigma_{oj}^2} \right]^2 \right.$$

$$\left. - \left[\left(\frac{m_{1j}\sigma_{oj}^2 - m_{oj}\sigma_{1j}^2}{\sigma_{1j}^2 - \sigma_{oj}^2} \right)^2 - \left(\frac{m_{oj}^2\sigma_{1j}^2 - m_{1j}^2\sigma_{oj}^2}{\sigma_{1j}^2 - \sigma_{oj}^2} \right) \right] \right\} \quad (5\text{-}19)$$

But, since it can be seen that the terms in the second bracket in (5-19) don't depend on r_j, they are just constants and can be separated from the exponent and lumped into a new threshold, η . So, after manipulating the constants, the test becomes

$$L(\underline{R}) = \prod_{j=1}^{N} \frac{\sigma_{oj}}{\sigma_{1j}} \exp \left| -\frac{1}{2} \sum_{j=1}^{N} \frac{(r_j - m_j)^2}{\sigma_j^2} \right| \quad (5\text{-}20)$$

where

$$m_j = \frac{m_{1j}\sigma_{oj}^2 - m_{oj}\sigma_{1j}^2}{\sigma_{1j}^2 - \sigma_{oj}^2} \quad (5\text{-}21)$$

and

$$\sigma_j^2 = \frac{\sigma_{oj}^2\sigma_{1j}^2}{|\sigma_{1j}^2 - \sigma_{oj}^2|} \quad (5\text{-}22)$$

We could then write the LRT for (5-20) in terms of a sufficient statistic as

$$L'(\underline{R}) = \sum_{j=1}^{N} \frac{(r_j - m_j)^2}{\sigma_j^2} \underset{H_o}{\overset{H_1}{\gtrless}} \gamma \quad (5\text{-}23)$$

If we rewrote (5-23) in terms of standard normal random variables, z_j, where

$$Z_j = \frac{r_j - m_j}{\sigma_j} \qquad (5\text{-}24)$$

then the LRT becomes

$$L'(\underline{R}) = \sum_{j=1}^{N} Z_j^2 \underset{H_o}{\overset{H_1}{\gtrless}} \gamma \qquad (5\text{-}25)$$

But, d is created by the sum of N random variables, each of which is the square of a standard normal random variable, and is, therefore, also a random variable which has a $\chi^2(N)$ distribution with N degrees of freedom (Ref. 8:196-198).

As will be seen, the result given by (5-25) will be used as the LRT for the classifier described in this paper while the distribution of d will be used both to establish the thresholds for the LRT's and to predict the performance of the classifier.

VI. Phoneme Decision Rule

In the previous chapter, the background of statistical hypothesis testing was outlined briefly. In the present chapter, a modification of the Neyman-Pearson criterion is used to derive an M-ary decision rule that is well suited to the objectives of this research. The Neyman-Pearson criterion was chosen for the task because of its flexibility. The objective of this chapter is to derive a decision rule which maximizes the average probability of phoneme detections while subject to some subjective or empirical constraint on the average probability of false classifications.

In one of the intermediate steps in the derivation, an assumption is made which, in theory, will result in the possibility of alternative hypotheses for a single observation. The assumption is that, unlike forced decision rules, the decision regions need not be mutually exclusive to one another. This is desirable since, for sounds that are easily confused for one another, we would like to let the syntactic processor handle the ambiguity rather than to force a decision.

Proceeding with the development of a decision rule, a gain function, G, is defined in terms of P_D, the average conditional probability of detection and P_{FC}, the average conditional p[illegible]ility of false classification.

$$G = P_D - \lambda \left[P_{FC} - k' \right] \qquad (6\text{-}1)$$

where

$$P_D = \frac{1}{M+1} \left[\int_{z_0} p(\underline{R}|H_o)\, d\underline{R} + \int_{z_1} p(\underline{R}|H_1)\, d\underline{R} + \ldots + \int_{z_M} p(\underline{R}|H_M)\, d\underline{R} \right] \qquad (6\text{-}2)$$

and

$$P_{FC} = \frac{1}{M+1} \left\{ \left[\int_{z_1} p(\underline{R}|H_o)\, d\underline{R} + \int_{z_2} p(\underline{R}|H_o)\, d\underline{R} + \ldots + \int_{z_M} p(\underline{R}|H_o)\, d\underline{R} \right] + \left[\int_{z_o} p(\underline{R}|H_1)\, d\underline{R} + \int_{z_2} p(\underline{R}|H_1)\, d\underline{R} + \ldots + \int_{z_M} p(\underline{R}|H_1)\, d\underline{R} \right] + \ldots + \left[\int_{z_o} p(\underline{R}|H_M)\, d\underline{R} + \int_{z_1} p(\underline{R}|H_1)\, d\underline{R} + \ldots + \int_{z_{M-1}} p(\underline{R}|H_M)\, d\underline{R} \right] \right\} \qquad (6\text{-}3)$$

Next, the terms of (6-2) are grouped with those of (6-3) having common regions of integration so that, after substitution, (6-1) becomes

$$G = \frac{1}{M+1} \int_{Z_o} \left[p(\underline{R}|H_o) - \lambda \sum_{\substack{i=0 \\ i\neq 0}}^{M} p(\underline{R}|H_i) \right] d\underline{R}$$

$$+ \quad \frac{1}{M+1} \int_{Z_1} \left[p(\underline{R}|H_1) - \lambda \sum_{\substack{i=0 \\ i\neq 1}}^{M} p(\underline{R}|H_i) \right] d\underline{R}$$

$$+ \dots + \frac{1}{M+1} \int_{Z_M} \left[p(\underline{R}|H_M) - \lambda \sum_{\substack{i=0 \\ i\neq M}}^{M} p(\underline{R}|H_i) \right] d\underline{R} \quad + \quad \lambda k' \qquad (6\text{-}4)$$

Now, in order to maximize G (the regions are mutually exclusive up to this point) for fixed λ, we should maximize the sum of the integrations simultaneously. However, if the requirement for mutually exclusive decision regions can be relaxed, G could be maximized by maximizing each integral in (6-4) independently of one another. This is done, as before, by assigning $\underline{R}$ to Z_k if and only if a positive contribution to the integrand results.

Considering only the kth integral in (6-4), the test is

$$p(\underline{R}|H_k) \underset{H_k'}{\overset{H_k}{\gtrless}} \lambda \sum_{\substack{i=0 \\ i\neq k}}^{M} p(\underline{R}|H_i) \qquad (6\text{-}5)$$

which is the rule for saying whether or not H_k should be among the set of alternative hypotheses when $\underline{R}$ is observed. Inequality (6-5) can be rewritten as

$$\sum_{\substack{i=0 \\ i\neq k}}^{M} \frac{p(\underline{R}|H_i)}{p(\underline{R}|H_k)} \underset{H_k}{\overset{H_k'}{\gtrless}} \frac{1}{\lambda} \triangleq \eta \qquad (6\text{-}6)$$

which is recognized as the sum of M LRT's compared against a single threshold.

Unfortunately, this result is not very convenient to implement for a near-real-time classifier. Taking the log of (6-6) would not result in a simple test either since the log of a sum is not equal to the sum of the logs.

We could, however, make a more restrictive test (with respect to the constraint on P_{FC}) with M pairwise LRT's. That is, P_{FC} will be lower if each pairwise LRT must support H_k for it to be in the set of alternatives. This can be easily proven using the Union Bound (Ref. 22:264-266) which says that the probability of an event made up of the finite union of subevents is upper bounded by the sum of probabilities of the subevents. Or, more precisely,

$$\Pr\left|\xi\right| \leq \sum_{\substack{i=0 \\ i\neq k}}^{M} \Pr\left|\xi_i\right| \tag{6-7}$$

where the subevents, ξ_i, are the events that the pairwise LRT's erroneously support H_k and ξ is the union of those subevents. This is made more apparent by writing (6-6) as

$$\sum_{\substack{i=0 \\ i\neq k}}^{M} \frac{p(\underline{R}|H_i)}{p(\underline{R}|H_k)} \underset{H_k}{\overset{H_k'}{\gtrless}} \sum_{\substack{i=0 \\ i\neq k}}^{M} \eta_{ik} \tag{6-8}$$

where each of the η_i's can be associated with the ith LRT. In the form of (6-7), this becomes

$$\Pr\left\{ \text{false classification} \mid H_k' \right\} \leq \sum_{\substack{i=0 \\ i \neq k}}^{M} \Pr\left\{ L_{ik}(\underline{R}) < \eta_{ik} \right\} \quad (6\text{-}9)$$

where

$$L_{ik}(\underline{R}) = \frac{p(\underline{R} \mid H_i)}{p(\underline{R} \mid H_k)} \quad (6\text{-}10)$$

Thus, as a result of using the stricter test, we have reduced (6-6) to M simultaneous LRT's where the i-k th LRT is given by (6-10).

As should be expected, though, we pay a penalty for implementing the simpler test. For example, after fixing the allowable P_{FC}, we will solve for the thresholds that will yield that value of P_{FC}. But, when implementing the classifier with those thresholds, the observed P_{FC} will be lower since the test is stricter. However, the observed P_D will also be lower since the thresholds will be smaller than they could have theoretically been. So, a loss in performance is the penalty we pay for a simpler test.

We can define P_{FC_k}, the average probability of false classification given that H_k was not true, from (6-9) where we assume that the equality holds. That is,

$$P_{FC_k} = \sum_{\substack{i=0 \\ i \neq k}}^{M} P_{FC_{ik}} \quad (6\text{-}11)$$

where

$$P_{FC_{ik}} = \Pr \left\{ L_{ik}(\underline{R}) \leq \eta_{ik} \right\}$$

$$= \int_{-\infty}^{\eta_{ik}} p\left(L_{ik}(\underline{R}) \mid H_k' \right) \, dL_{ik}(\underline{R}) \qquad (6\text{-}12)$$

is the chance that the i-k th LRT will not exceed its associated threshold when hypothesis K was not true. In other words, when H_k is not true, each LRT should exceed its associated threshold. Otherwise, H_k might be erroneously accepted as an alternative. This potential error is denoted as $P_{FC_{ik}}$ and can be calculated using (6-12).

From (5-20) we can see that L($\underline{R}$) is just a function of the random vector $\underline{R}$. Recognizing that the quantity L($\underline{R}$) is itself a random variable, then (6-12) is simply the cumulative distribution function (cdf) for the random variable L($\underline{R}$).

Alternatively, a sufficient statistic L'($\underline{R}$) (another random variable), given by (5-25) could be used to calculate the $P_{FC_{ik}}$'s. Since L'($\underline{R}$) is $\chi^2(N)$ distributed, (6-12) becomes

$$P_{FC_{ik}} = \int_0^{\gamma_{ik}} \frac{1}{2^{\frac{N}{2}} \Gamma\left(\frac{N}{2}\right)} \, x^{\left(\frac{N-2}{2}\right)} \exp \left| -\frac{1}{2} x \right| dx \qquad (6\text{-}13)$$

where

$$\gamma_{ik} = -2 \left\{ \log \eta_{ik} - \sum_{j=1}^{N} \log \left[\frac{\sigma_{ij}}{\sigma_{kj}} \right] + \right.$$

$$+ \sum_{j=1}^{N} \frac{\sigma_{ij}^2 - \sigma_{kj}^2}{\sigma_{ij}^2 \sigma_{kj}^2} \left[\left(\frac{m_{ij}\sigma_{kj}^2 - m_{kj}\sigma_{ij}^2}{\sigma_{ij}^2 - \sigma_{kj}^2} \right) - \left(\frac{m_{kj}^2\sigma_{ij}^2 - m_{ij}^2\sigma_{kj}^2}{\sigma_{ij}^2 - \sigma_{kj}^2} \right) \right] \Bigg\} \tag{6-14}$$

In addition to having M of the equations given by (6-13) for each of the M+1 hypotheses, from (6-8) it can be seen that

$$\sum_{\substack{i=0 \\ i \neq k}}^{M} \eta_{ik} = \eta \qquad k=0,1,\ldots,M \tag{6-15}$$

must be satisfied.

The thresholds, η_{ik}, can be solved for using (6-11) and (6-13) if we know the value of P_{FC_k}. But without prior evidence to the contrary, it would be reasonable to assume that we can tolerate equal average errors for each phoneme. Then, recalling the definition of P_{FC} in (6-3), we can say that $P_{FC_k} = P_{FC}$.

At this point, there are several considerations that must be discussed so that a solution algorithm will be able to converge on a feasible solution.

From equations (6-11), (6-13), and (6-15), there are 2M+1 basic variables (the $P_{FC_{ik}}$'s and η_{ik}'s) in the solution while, from equations (6-13) and (6-15), there are only M+1 equations. This indicates that multiple solutions probably exist. Therefore, in order to produce a unique solution, we will also assume that each of the thresholds in (6-15) are equal. Under this assumption, then,

$$\eta_{ik} = \eta/M \qquad i=0\ldots M, i\neq k \qquad (6\text{-}16)$$

While satisfying (6-15) (and, hence, eliminating one of the equations) this assumption also eliminates M variables (the η_{ik}'s) from the basis. However, since η must still be solved for, it must now enter the solution basis. Therefore, with M+1 basic variables and M+1 equations, a unique solution may be found if none of the equations are dependent.

Figure 13 summarizes the algorithm which solves for the decision thresholds. This algorithm is implemented in a FORTRAN program given in Appendix A. Notice that the algorithm integrates over increasingly larger intervals (by increasing η) until the sum of the $P_{FC_{ik}}$'s just exceeds the specified P_{FC}. At that point, each of the M thresholds, γ_{ik}, for a single hypothesis, k, become known. This procedure is repeated independently for each of the M+1 hypotheses. The starting values for γ are those calculated by (6-14) with η equalling some small number.

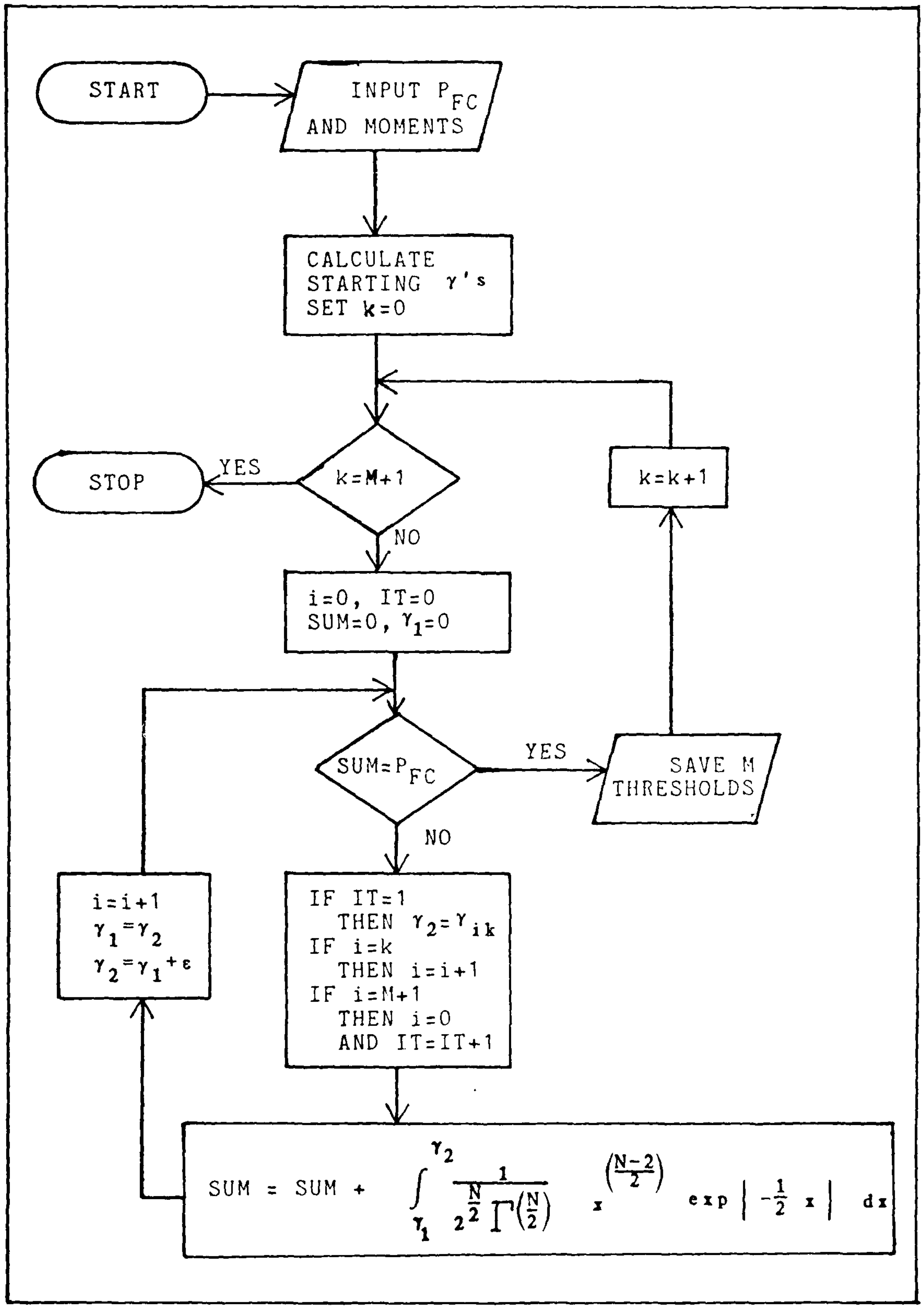
START
INPUT P_{FC} AND MOMENTS
CALCULATE STARTING γ's SET $k=0$
$k=M+1$
YES
STOP
NO
$k=k+1$
$i=0$, $IT=0$
$SUM=0$, $\gamma_1=0$
$SUM=P_{FC}$
YES
SAVE M THRESHOLDS
NO
$i=i+1$
$\gamma_1=\gamma_2$
$\gamma_2=\gamma_1+\varepsilon$
IF $IT=1$ THEN $\gamma_2=\gamma_{ik}$
IF $i=k$ THEN $i=i+1$
IF $i=M+1$ THEN $i=0$ AND $IT=IT+1$
$SUM = SUM + \int_{\gamma_1}^{\gamma_2} \frac{1}{2^{\frac{N}{2}}\Gamma\left(\frac{N}{2}\right)} x^{\left(\frac{N-2}{2}\right)} \exp\left|-\frac{1}{2}x\right| dx$

Figure 13. Decision Thresholds Solution Algorithm

VII. Results and Conclusions

The desired result of this research was the development of an efficient phoneme decision rule that permits the use of a syntactic processor to perform error correction. Limited success was achieved in satisfying this goal.

An algorithm for the integration of the phoneme decision rule and a backtracking parser was described in Chapter IV. The algorithm as described there has been only partially implemented.

The major component which is missing in this implementation is the data base. At the present stage of implementation, the data base consists simply of disk files which store the data in a readable format. That is, the manipulation of the data files must presently be done manually. Although much time and effort was expended in attempting to design an efficient data base, limitations in time dictated that efforts should be redirected to other remaining problems.

Also, due to limitations in time, testing of the phoneme decision rule developed in Chapter VI was not possible. Although it would have been possible to generate the LPC st tistics for a few utterances and manually feed them into the threshold solution program, it was decided that more benefit would come from beginning the development of an interactive program that would speed the entire testing process and provide a simple interface to the user. The program, CLASSIFY, is the result of this decision. One task remains in the development of this program: designing the interface to a data base containing the lexical entries.

Considerably greater success was made in the theoretical development of the phoneme decision rule. A new M-ary decision rule was derived based on the assumption that non-mutually exclusive decision regions were desired. The result of this assumption is that alternative phonemes can exist for each observation.

The development of the phoneme decision rule was made with a strong emphasis on efficiency and adaptibility to changes in performance specifications. Because the decision rule uses LPC parameters as a feature set and makes its decision based on an efficient likelihocd ratio test (LRT), it is believed that the algorithm has a great potential for real-time implementation.

In addition, this decision rule interfaces quite naturally with a non-deterministic syntactic error corrector such as a backtracking parser.

In conclusion, therefore, it is held that the primary research objective, that is, the development of a unified approach for integrating a feature extractor and a syntactic error corrector into a CSR system, has been satisfied. Unfortunately, the secondary objective, that of providing an experimental system implementation to test the algorithm, has only been partially satisfied.

VIII. Recommendations

Although the objectives of the research have only been partially satisfied, recommendations for future work are still warranted. The recommendations made here are made with the intention of investigating the utility of a system which implements the phoneme decision rule and backtracking parser discussed in this paper.

The first recommendation is that an efficient lexical retrieval system be implemented. One of the factors that must be considered is the computer language in which the retrieval system is written. This is because many of the difficulties encountered here in attempts to implement a data base were due to the poor disk handling and data structure capabilities of the PASCAL and ALGOL languages respectively that were available at this computer installation. FORTRAN was ruled out entirely because string manipulation is almost impossible in that language. If PASCAL is to be used, it is strongly suggested that a better compiler be acquired.

Once a language is decided upon, it must also be decided which attributes of the lexical entries will be included in the data base and how they will be linked to the entries. Two of the possible attributes might be the grammatical catagories of which the entry is a member (such as verb, noun, etc.) and the phonological spellings of the entry.

Once the lexical retrieval system has been implemented, the testing of the system will be made easy using the interactive program, CLASSIFY. The testing should proceed by generating a set of thresholds for a single speaker through a series of training sessions. Then, attempts at recognition of sentences containing words in the lexicon should be made, noting the actual performance. This should be repeated for a number of specified performance levels (P_{FC}). The corresponding recognition results should also be noted along with the computation times.

Then, finally, based on the results of these recommended tests, the potential utility of the CSR algorithm described in this paper can be assessed.

Bibliography

1. Cohen, Paul S. and Robert L. Mercer. "The Phonological Component of an Automatic Speech-Recognition System," Speech Recognition, edited by D. Raj Reddy. New York: Academic Press, Inc., 1974.

2. Denn, Morton M. Optimization by Variational Methods. New York: McGraw-Hill, 1969.

3. Flanagan, James L. Speech, Analysis, Synthesis and Perception. New York: Academic Press, 1965.

4. Forsythe, George E., et al. Computer Methods for Mathematical Computations. New Jersey: Prentice-Hall, 1977.

5. Itakura, Fumitada. "Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Transactions on Acoustics, Speech, and Signal Processing, 23: 67-72 (1975).

6. Kashap, R. L. "Syntactic Decision Rules for Recognition of Spoken Words and Phrases Using a Stochastic Automation," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2:154-163 (April 79).

7. Markel, J. D. and A. H. Gray, Jr. Linear Prediction of Speech. New York: Springer-Verlag, 1976.

8. Meyer, Paul L. Introductory Probability and Statistical Applications. Massachusetts: Addison-Wesley, 1968.

9. Narashima, W. J., K. Shenoi, and A. M. Peterson. "A Hilbert Space Approach to Linear Predictive Analysis of Speech Signals," Interim Report, Government Contract # N00014-67-A-00112-0044, February 1974.

10. Newman, William M. and Robert F. Sproull. Principles of Interactive Computer Graphics. New York: McGraw-Hill, 1979.

11. Neyman, J. and E.S. Pearson. "On the Problem of the Most Efficient Tests of Statistical Hypotheses," Joint Statistical Papers. Berkeley and Los Angelos: University of California Press, 1967.

12. Oppenheim, Alan V. and Ronald W. Schafer. Digital Signal Processing. New Jersey: Prentice-Hall, 1975.

13. Rabiner, Lawrence R. and Bernard Gold. Theory and Application of Digital Signal Processing. New Jersey:

Prentice-Hall, 1975.

14. Srinath, M.D. and P.K. Rajasekaran. An Introduction to Statistical Signal Processing With Applications. New York: Wileys 1979.

15. Van Trees, Harry L. Detection, Estimation, and Modulation Theory. New York: Wiley, 1968.

16. Wakita, H. "Direct Estimation of the Vocal Tract Shape by Inverse Filtering of Acoustic Speech Waveforms," IEEE Transactions, 21: 417-427 (1973).

17. Weinstein, Clifford J., et al. "A System for Acoustic-Phonetic Analysis of Continuous Speech," IEEE Transactions on Acoustics, Speech, and Signal Processing, 23: 54-67 (February 1975).

18. White, George M. "Speech Recognition: a Tutorial Overview," Computer, 9: 40-53 (May 1976).

19. Willsky, Alan S. Digital Signal Processing and Control and Estimation Theory. Cambridge, Massachusetts: MIT Press, 1979.

20. Wirth, Nicklaus. Algorithms Data Structures = Programs. New Jersey: Prentice Hall, 1976.

21. Woods, William A. "Syntax, Semantics, and Speech," Speech Recognition, edited by D. Raj Reddy. New York: Academic Press, Inc., 1974.

22. Wozencraft, John M. and Irwin Mark Jacobs. Principles of Communication Engineering. New York: Wiley, 1965.

Appendix A

Program Listings

```
C
C  PROGRAM CLASSIFY.5
C
C  THIS PROGRAM ALLOWS INTERACTIVE CLASSIFICATION OF SPEECH
C  SEGMENTS BY DISPLAYING SPECTROGRAMS ON THE TEKTRONIX IN
C  GRAY SCALE OF THE LPC MODEL SPECTRA.
C
C  SUBROUTINES USED: SCARRAY, SPCTRM,TDELAY,ERS,TONE
C
C  AUTHOR: ROBERT B. TAYLOR
C  DATE:   15 NOV.    1980
C
C
      PARAMETER M = 12, N = 50, IXOFF = 0, IYOFF = 1
C
      DIMENSION AC(M),AMAG(90,N),TEMP(N)
C
      OPEN 1, "LPC.AC"
      OPEN 2, "LPC.RC"
C
      BIG = -1.0 E +55
      SMALL = 1.0 E +55
      MIN = 1
      MAX = 1
C
C  GET LPC COEFFICIENTS
C
C
      CALL FGTIME(IHR1,IMIN1,ISEC1)
      TYPE "INPUT START TIME -        ",IHR1,":",IMIN1,":",ISEC1
C
      DO 10 I = 1,90
        READ FREE (1) (AMAG(I,J), J=1,M)
   10 CONTINUE
C
      CALL FGTIME(IHR2,IMIN2,ISEC2)
      TYPE "INPUT STOP TIME -         ",IHR2,":",IMIN2,":",ISEC2
C
      TYPE "CALCULATING SPECTRUM..."
C
C  CALCULATE THE MAGNITUDE SPECTRUM OF THE ENTIRE 3 SECONDS
C  OF SPEECH USING THE LPC PREDICTOR COEFFICIENTS
C
      DO 40 I = 1,90
        DO 20 J = 1,M
          AC(J) = AMAG(I,J)
   20   CONTINUE
        CALL SPCTRM(AC,M,TEMP,N)
        CALL SCARRAY(TEMP,N,AMIN,IMIN,AMAX,IMAX)
        IF (AMIN .LT. SMALL) SMALL = AMIN
        IF (AMIN .LT. SMALL) MIN = I
        IF (AMAX .GT. BIG) MAX = I
        IF (AMAX .GT. BIG) BIG = AMAX
        DO 30 J = 1,N
```

```
X             IF (TEMP(J) .GT. 15.) TEMP(J) = 15.
            AMAG(I,J) = TEMP(J)
   30   CONTINUE
   40 CONTINUE
C
      CALL FGTIME(IHR,IMIN,ISEC)
      TYPE "END CALCULATE SPECTRUM - ",IHR,":",IMIN,":",ISEC
C
C
C  GET THE NEXT COMMAND
C
   45 TYPE " "
      TYPE "THE LARGEST MAGNITUDE IS - ",BIG
      TYPE "     - ITS LOCATION IS IN SEGMENT ",MIN
      TYPE " "
      TYPE "THE SMALLEST MAGNITUDE IS - ",SMALL
      TYPE "     - ITS LOCATION IS IN SEGMENT ",MAX
      TYPE " "
      TYPE "WHICH SEGMENT TO EXAMINE? (1-90) "
      TYPE "TYPE 99 TO VIEW ALL SEGMENTS IN A SPECTROGRAM"
      TYPE "- OR  0 TO EXIT - "
      ACCEPT "- OR 1-90 TO VIEW A SINGLE TIME SEGMENT ",IANS
      IF (IANS .EQ. 0) STOP
      IF (IANS .EQ. 99) GO TO 60
      IF (IANS .LT. 0 .OR. IANS .GT. 99) STOP
      ACCEPT "WISH TO SPECIFY SCALES? (1 FOR YES,0 FOR NO) ",
     +       IANS2
      IF (IANS2 .NE. 1) GO TO 48
      ACCEPT "LOW VALUE - ",SMIN
      ACCEPT "HIGH VALUE - ",SMAX
      IFSCL = 1
      GO TO 49
   48 IFSCL = 0
C
   49 DO 50 J = 1,N
        TEMP(J) = AMAG(IANS,J)
   50 CONTINUE
      CALL GRPH2S("MODEL SPECTRUM",1,TEMP,
      TEMP,N,1,SMIN,SMAX,IFSCL)
      PAUSE IN-CLASSIFY
      GO TO 45
C
C  PLOT THE SPECTROGRAM (LPC MODEL SPECTRUM) IN
C  GRAY TONE SCALE (10 LEVELS OF GRAY - 0 THROUGH 9)
C
C  SUGGESTED PARAMETERS -
C        CONTRAST = 7.5, BIAS = -5.
C
   60 ACCEPT "INPUT THE DESIRED CONTRAST LEVEL ",CONTR
      ACCEPT "INPUT THE DESIRED BIAS ",BIAS
      DO 100 I = 1,90
        DO 80 J = 1,N
          MAG = AMAG(I,J) * CONTR + BIAS
          CALL TONE(I+IXOFF,J+IYOFF,MAG)
   80   CONTINUE
  100 CONTINUE
C
C  GO TO HOME AND PUT UP CROSSHAIR,
C  ACCEPT TWO-CHARACTER CODE FOR PHONEME
C
      CALL TPLOT(0,1000,2)
      CALL  PCHAR(31,IER)
      CALL CURSOR(IX,IY,ICHAR1)
      IXLEFT = IX / 9
      CALL TPLOT(IX,480,2)
      CALL TPLOT(IX,500,1)
      CALL CURSOR(IX,IY,ICHAR2)
      IXRIGHT = IX / 9 + 1
      CALL TPLOT(IX,480,2)
      CALL TPLOT(IX,500,1)
      IXMID = (IXRIGHT + IXLEFT) * 9 / 2
      CALL TPLOT(IXMID,475,2)
      CALL PCHAR(37K,IER)
      CALL PCHAR(ICHAR1,IER)
      CALL PCHAR(ICHAR2,IER)
      CALL TPLOT(10,780,2)
      CALL PCHAR(37K,IER)
      ICHAR = ICHAR1*256 + ICHAR2
      TYPE IXLEFT,IXRIGHT
C
C
C  ALTHOUGH NOT YET IMPLEMENTED, THE OUTPUTS OF CLASSIFY
C  WILL BE ALL THE VALUES OF ICHAR AND THE CORRESPONDING
C  START AND STOP SEGMENT NUMBERS
C
C
      ACCEPT "TYPE 1 TO GO BACK TO MODE 1,
     +     0 TO STOP, OR 2 TO CONTINUE CLASSIFYING",INUM
      IF (INUM .EQ. 1) GO TO 45
      IF (INUM .EQ. 2) GO TO 100
      CALL RESET
      STOP
      END
      SUBROUTINE CURSOR(IX,IY,ICHAR)
C
C
C  THIS SUBROUTINE PUTS UP THE CROSSHAIRS ON THE TEKTRONIX
C  SCREEN AND RETURNS THE CURRENT X AND Y LOCATION
C  PLUS THE KEY DEPRESSED.
C  RETURN IS IN ALPHA MODE.
C
C
C  ARGUMENTS -
C
C     IX - THE X LOCATION IN TEK POINTS (0 - 1024)
C     IY - THE Y LOCATION IN TEK POINTS (0 - 767)
C     ICHAR - THE ASCII CODE FOR THE KEY DEPRESSED
C
C  AUTHOR - ROBERT B. TAYLOR
C  DATE -   14 NOV.  1980
C
      DIMENSION IBYTE(5)
C
C  BEFORE PUTTING UP CROSSHAIRS, TEK MUST BE IN ALPHA MODE
C
      CALL PCHAR(37K,IER)
C
C  SEND AN ESC-SUB TO PUT UP CROSSHAIRS
C  AND GET X AND Y LOCATIONS AND KEY DEPRESSED
C
      CALL PCHAR(33K,IER)
      CALL PCHAR(32K,IER)
      DO 10 I = 1,5
        CALL GCHAR(IBYTE(I),IER)
        CALL CHECK(IER)
   10 CONTINUE
C
C  COMPUTE X AND Y COORDINATES
C
      IX = 32*(IBYTE(2) - 32) + (IBYTE(3) - 32)
      IY = 32*(IBYTE(4) - 32) + (IBYTE(5) - 32)
      ICHAR = IBYTE(1)
      CALL PCHAR(37K,IER)
      CALL CHECK(IER)
      RETURN
      END
```

```
C
C   PROGRAM MAKELPC.5
C
C
C   PARAMETERS
C
C      M - THE ORDER OF THE LPC FILTER
C      NSAMP - THE NUMBER OF SPEECH SAMPLES PER FRAME
C
C
C   AUTHOR - ROBERT B. TAYLOR
C         15 NOV.  1980
C
C
C   SUBROUTINES CALLED - AUTO
C
       PARAMETER  M = 12, NSAMP = 256
C
       DIMENSION X(NSAMP),TEMP(NSAMP),IARRAY(NSAMP),RC(M),AC(M)
C
       OPEN 1,"DSPEECH"
       OPEN 2,"LPC.AC"
       OPEN 3,"LPC.RC"
C
C   THERE ARE 90 RECORD BLOCKS IN "DSPEECH" FOR 8KHZ SAMPLE RATE
C
       DO 100 I = 1,90
         CALL RDBLK(1,I,IARRAY,1,IER)
         DO 50 J = 1,NSAMP
           X(J) = IARRAY(J)
   50    CONTINUE
       CALL AUTO(NSAMP,X,M,AC,ALPHA,RC)
       WRITE FREE (2) (AC(J), J=1,M)
       WRITE FREE (3) (RC(J), J=1,M)
  100  CONTINUE
       CALL RESET
       STOP
       END
```

```
C
C   PROGRAM PREEMPHASIZE.5
C
C
C   THIS PROGRAM PERFORMS SPECTRAL PRE-EMPHASIS AT
C   6 dB / OCTAVE USING A DIGITAL FILTER OF THE FORM:
C
C      1 - Z**(-1)
C
      DIMENSION IARRAY(256)
C
      OPEN 1,"DSPEECH"
      OPEN 2,"DSPEECH1"
      ITEMP = 0
      DO 100 I = 1,90
        CALL RDBLK(2,I,IARRAY,1,IER)
        ITEMP2 = IARRAY(256)
        DO 50 J = 0,254
          IARRAY(256-J) = IARRAY(256-J) - IARRAY(256-J-1)
   50   CONTINUE
        IARRAY(1) = IARRAY(1) - ITEMP
        ITEMP = ITEMP2
        CALL WRBLK(1,I,IARRAY,1,IER)
  100 CONTINUE
      CALL RESET
      STOP
      END
```

```
C
C
      SUBROUTINE QUANC8(FUN,A,B,ABSERR,RELERR,
     +       RESULT,ERREST,NOFUN,FLAG)
C
C  THIS SUBROUTINE IS DUE TO FORSYTHE, ET AL. (REF. 4)
C
C
      REAL FUN,A,B,ABSERR,RELERR,RESULT,ERREST,FLAG
      INTEGER NOFUN
C
C  ESTIMATE THE INTEGRAL OF FUN(X) FROM A TO B
C  TO A USER PROVIDED TOLERANCE.
C  AN AUTOMATIC ADAPTIVE ROUTINE BASED ON
C  THE 8-PANEL NEWTON-COTES RULE.
C
C  INPUT..
C
C  FUN   THE NAME OF THE INTEGRAND FUNCTION SUBPROGRAM FUN(X).
C  A  THE LOWER LIMIT OF INTEGRATION.
C  B  THE UPPER LIMIT OF INTEGRATION.(B MAY BE LESS THAN A.)
C  RELERR   ARELATIVE ERROR TOLERANCE. (SHOULD BE NON-NEGATIVE)
C  ABSERR   AN ABSOLUTE ERROR TOLERANCE.
C             (SHOULD BE NON-NEGATIVE)
C
C  OUTPUT..
C
C  RESULT   AN APPROXIMATION TO THE INTEGRAL HOPEFULLY
C        SATISFYING
C        THE LEAST STRINGENT OF THE TWO ERROR TOLERANCES.
C  ERREST   AN ESTIMATE OF THE MAGNITUDE OF THE ACTUAL ERROR.
C  NOFUN THE NUMBER OF FUNCTION VALUES USED IN CALCULATION
C           OF RESULT.
C  FLAG  A RELIABILITY INDICATOR.  IF FLAG IS ZERO, THEN RESULT
C        PROBABLY SATISFIES THE ERROR TOLERANCE.  IF FLAG IS
C        XXX.YYY, THEN XXX = THE NUMBER OF INTERVALS WHICH HAVE
C        NOT CONVERGED AND O.YYY = THE FRACTION OF THE INTERVAL
C        LEFT TO DO WHEN THE LIMIT ON NOFUN WAS APPROACHED.
C
      REAL WO,W1,W2,W3,W4,AREA,XO,FO,STONE,STEP,COR11,TEMP
      REAL QPREV,QNOW,QDIFF,QLEFT,ESTERR,TOLERR
      REAL QRIGHT(31),F(16),X(16),FSAVE(8,30),XSAVE(8,30)
      INTEGER LEVMIN,LEVMAX,LEVOUT,NOMAX,NOFIN,LEV,NIM,I,J
C
C  ***  STAGE 1 ***  GENERAL INITIALIZATION
C  SET CONSTANTS.
C
      LEVMIN = 1
      LEVMAX = 30
      LEVOUT = 6
      NOMAX = 5000
      NOFIN = NOMAX - 8*(LEVMAX-LEVOUT+2**(LEVOUT+1))
C
C  TROUBLE WHEN NOFUN REACHES NOFIN
```

```
C
      W0 = 3956.0 / 14175.0
      W1 = 23552.0 /14175.0
      W2 = -3712.0 / 14175.0
      W3 = 41984.0 / 14175.0
      W4 = -18160.0 / 14175.0
C
C  INITIALIZE RUNNING SUMS TO ZERO.
C
      FLAG = 0.0
      RESULT = 0.0
      CORR11 = 0.0
      ERREST = 0.0
      AREA = 0.0
      NOFUN = 0
      IF (A .EQ. B) RETURN
C
C  ***  STAGE 2 ***  INITIALIZATION FOR FIRST INTERVAL
C
      LEV = 0
      NIM = 1
      X0 = A
      X(16) = B
      QPREV = 0.0
      F0 = FUN(X0)
      STONE = (B - A) / 16.0
      X(8) = (X0 + X(16)) / 2.0
      X(4) = (X0 + X(8)) / 2.0
      X(12) = (X(8) + X(16)) / 2.0
      X(2) = (X0 + X(4)) / 2.0
      X(6) = (X(4) + X(8)) / 2.0
      X(10) = (X(8) + X(12)) / 2.0
      X(14) = (X(12) + X(16)) / 2.0
      DO 25 J = 2,16,2
        F(J) = FUN(X(J))
   25 CONTINUE
      NOFUN = 9
C
C  ***  STAGE 3 ***  CENTRAL CALCULATION
C  REQUIRES QPREV,X0,X2,X4,...,X16,F0,F2,F4,...,F16.
C  CALCULATES X1,X3,...,X15,F1,F3,...,F15,
C            QLEFT,QRIGHT,QNOW,QDIFF,AREA.
C
   30 X(1) = (X0 + X(2)) / 2.0
      F(1) = FUN(X(1))
      DO 35 J = 3,15,2
        X(J) = (X(J-1) + X(J+1)) / 2.0
        F(J) = FUN(X(J))
   35 CONTINUE
      NOFUN = NOFUN + 8
      STEP = (X(16) - X0) / 16.0
      QLEFT = (W0*(F0 + F(8)) + W1*(F(1)+F(7)) + W2*(F(2)+F(16))
     1      + W3*(F(3)+F(5)) + W4*F(4)) * STEP
      QRIGHT(LEV+1) = (W0*(F(8)+F(16))+W1*(F(9)+F(15))
     +               + W2*(F(10)+F(14))
     +               + W3*(F(11)+F(13)) + W4*F(12)) * STEP
      QNOW = QLEFT + QRIGHT(LEV+1)
      QDIFF = QNOW - QPREV
      AREA = AREA + QDIFF
C
C  ***  STAGE 4 *** INTERVAL CONVERGENCE TEST
C
      ESTERR = ABS(QDIFF) / 1023.0
      TOLERR = AMAX1(ABSERR,RELERR*ABS(AREA)) * (STEP/STONE)
      IF (LEV .LT. LEVMIN) GO TO 50
      IF (LEV .GE. LEVMAX) GO TO 62
      IF (NOFUN .GT. NOFIN) GO TO 60
      IF (ESTERR .LE. TOLERR) GO TO 70
C
C  ***  STAGE 5 ***  NO CONVERGENCE
C  LOCATE NEXT INTERVAL.
C
   50 NIM = 2*NIM
      LEV = LEV+1
C
C  STORE RIGHT HAND ELEMENTS FOR FUTURE USE.
C
      DO 52 I = 1,8
        FSAVE(I,LEV) = F(I+8)
        XSAVE(I,LEV) = X(I+8)
   52 CONTINUE
C
C  ASSEMBLE LEFT HAND ELEMENTS FOR IMMEDIATE USE.
C
      QPREV = QLEFT
      DO 55 I = 1,8
        J = -I
        F(2*J+18) = F(J+9)
        X(2*J+18) = X(J+9)
   55 CONTINUE
      GO TO 30
C
C  ***  STAGE 6 ***  TROUBLE SECTION
C  NUMBER OF FUNCTION VALUES IS ABOUT TO EXCEED LIMIT
C
   60 NOFIN = 2*NOFIN
      LEVMAX = LEVOUT
      FLAG = FLAG + (B - X0) / (B - A)
      GO TO 70
C
C  CURRENT LEVEL IS LEVMAX
C
   62 FLAG = FLAG + 1.0
C
C  ***  STAGE 7 ***  INTERVAL CONVERGED
C  ADD CONTRIBUTIONS INTO RUNNING SUMS.
C
   70 RESULT = RESULT + QNOW
```

```
      ERREST = ERREST + ESTERR
      COR11 = COR11 + QDIFF / 1023.0
C
C  LOCATE NEXT INTERVAL.
C
   72 IF (NIM .EQ. 2*(NIM/2)) GO TO 75
      NIM = NIM/2
      LEV = LEV-1
      GO TO 72
   75 NIM = NIM + 1
      IF (LEV .LE. 0) GO TO 80
C
C  ASSEMBLE ELEMENTS REQUIRED FOR THE NEXT INTERVAL.
C
      QPREV = QRIGHT(LEV)
      X0 = X(16)
      F0 = F(16)
      DO 78 I = 1,8
        F(2*I) = FSAVE(I,LEV)
        X(2*I) = XSAVE(I,LEV)
   78 CONTINUE
      GO TO 30
C
C  ***  STAGE 8 ***  FINALIZE AND RETURN
C
   80 RESULT = RESULT + COR11
C
C  MAKE SURE ERREST NOT LESS THAN ROUNDOFF LEVEL.
C
      IF (ERREST .EQ. 0.0) RETURN
   82 TEMP = ABS(RESULT) + ERREST
      IF (TEMP .NE. ABS(RESULT)) RETURN
      ERREST = 2.0*ERREST
      GO TO 82
      END
```

```
C
C-----------------------------------------------------------------
C
C     REPLAY ORIGINAL SPEECH ROUTINE
C
C-----------------------------------------------------------------
C
C
      DIMENSION INPUT(256)
C
      CALL DFILW("DSPOUT",IER)
      IF((IER.NE.1).AND.(IER.NE.13)) GO TO 900
      CALL FOPEN(3,"DSPOUT",512)
      CALL OPEN(7,"DSPEECH",2,IER,512)
      IF (IER.NE.1) GO TO 920
C
      ACCEPT "TYPE IN STARTING AND ENDING SEGMENTS",ISTART,IEND
      IF (ISTART.LE.1) ISTART = 1
      IF (IEND.GT.90) IEND = 90
      DO 50 I=1,256
        INPUT(I) = 0
   50 CONTINUE
      IF (ISTART.EQ.1) GO TO 200
      IS1 = ISTART - 1
      DO 100 I = 1,IS1
        CALL WRBLK(3,I,INPUT,1,IER)
        IF (IER.NE.1) GO TO 940
  100 CONTINUE
  200 DO 300 I = ISTART,IEND
        CALL RDBLK(7,I,INPUT,1,IER)
        IF (IER.NE.1) GO TO 940
        CALL WRBLK(3,I,INPUT,1,IER)
        IF (IER.NE.1) GO TO 960
  300 CONTINUE
      IF (IEND.EQ.90) GO TO 1000
      IE1 = IEND + 1
      DO 400 I = 1,256
        INPUT(I) = 0
  400 CONTINUE
      DO 500 I = IE1,90
        CALL WRBLK(3,I,INPUT,1,IER)
        IF (IER.NE.1) GO TO 960
  500 CONTINUE
C
      GO TO 1000
  900 WRITE(10,910) IER
  910 FORMAT(" FILE DELETING ERROR CODE - ",I3)
      GO TO 1000
  920 WRITE(10,930) IER
  930 FORMAT(" FILE OPENING ERROR CODE - ",I3)
      GO TO 1000
  940 WRITE(10,950) IER
  950 FORMAT(" RDBLK ERROR CODE - ",I3)
      GO TO 1000
```

```
960  WRITE(10,970) IER
970  FORMAT(" WRBLK ERROR CODE - ",I3)
1000   CALL RESET
     STOP
     END
```

```
      SUBROUTINE SPCTRM(ARRAY,M,SPEC,N)
C
C
C   SUBROUTINE SPCTRM COMPUTES THE MAGNITUDE SPECTRUM OF THE
C   SPEECH MODEL FROM THE PREDICTOR COEFFICIENTS, A.
C
C   ARGUMENTS:
C
C      ARRAY - THE PREDICTOR COEFFICIENTS (INPUT)
C      M - ORDER OF THE PREDICTOR FILTER
C      SPEC - THE MAGNITUDE SPECTRUM (OUTPUT)
C      N - THE NUMBER OF POINTS IN THE SPECTRUM
C             TO BE COMPUTED (INPUT)
C
C   AUTHOR: ROBERT B. TAYLOR
C   DATE:    15 NOV.     1980
C
C
      DIMENSION ARRAY(1),SPEC(200)
      COMPLEX Z,DENOM,ARG
C
      PI = 3.1459263
      DO 50 I = 1,N
        DENOM = (1.0,0.0)
        ARG = CMPLX(0.,PI*FLOAT(I-1)/FLOAT(N))
        DO 40 J = 2,M
          Z = CEXP(ARG*FLOAT(J))
          DENOM = ARRAY(J)*Z + DENOM
   40   CONTINUE
        SPEC(I) = 1./CABS(DENOM)
   50 CONTINUE
      RETURN
      END
```

```
C
C  SUBROUTINE TONE
C
C     JX - X POSITION OF DOT (FROM 0 TO 170)
C     JY - Y POSITION OF DOT (FROM 0 TO 128)
C     JZ - INTENSITY FROM 0 TO 9
C
C     THE HALFTONE PATTERNS GENERATED ARE THOSE
C     DESCRIBED IN ROBERT F. SPROULL'S "PRINCIPLES
C     OF INTERACTIVE COMPUTER GRAPHICS", PP. 225-27.
C
C     6  7  8
C     3  4  5  THE NUMBERS REPRESENT BIT POSITIONS IN KODE
C     0  1  2
C
      SUBROUTINE TONE (JX,JY,JZ)
      DIMENSION KODE(9)
      DATA KODE(1),KODE(2),KODE(3)/20K,60K,260K/
      DATA KODE(4),KODE(5),KODE(6)/270K,272K,273K/
      DATA KODE(7),KODE(8),KODE(9)/673K,773K,777K/
      IF(JZ .LE. 0) RETURN
      IF(JZ .GT. 9) JZ = 9
      DO 100 KY = 0,2
        IY = 5*(3*JY + KY)
        DO 50 KX = 0,2
          IF( ITEST(KODE(JZ),3*KY+KX) .EQ. 0) GO TO 50
          IX = 5*(3*JX + KX)
          CALL CHR(IX,IY,0)
          CALL CHR(IX,IY,1)
 50     CONTINUE
100   CONTINUE
      RETURN
      END
      REAL FUNCTION FUN(X)
C
C  THIS FUNCTION IS THE CHI-SQUARED DENSITY
C  FOR N=12 CORRESPONDING TO THE NUMBER OF
C  COMPONENTS IN THE LPC VECTOR
C
C  IT IS CALCULATED USING THE INTEGRAND OF
C  EQN. (6-3)
C
C  GAMMA6 IS THE GAMMA FCN. WITH AN ARGUMENT OF 6
C  THIS GIVES GAMMA6 = 5!
C
      GAMMA6 = 120
      FUN = 1./(2.**6 * GAMMA6)
      FUN = FUN * X**5 * EXP(-0.5 * X)
      RETURN
      END
```

VITA

Robert B. Taylor was born on 29 September 1954 in Cherry Point, North Carolina. He graduated from high school in 1973 and worked as an electrician for over a year. He then enlisted in the USAF and served as an Automatic Tracking Radar Repairman. With one and a half years in service, he was selected for an Air Force commission through the Airman Education and Commissioning Program. He graduated from Lehigh University with the degree of Bachelor of Science in Electrical Engineering in May 1979. Upon graduation, he was commissioned and selected to attend the School of Engineering, Air Force Institute of Technology.

Permanent address: Box 644 River Road
Upper Black Eddy, Pennsylvania
18972

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/80D-45	2. GOVT ACCESSION NO. AD-A100 807	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AUTOMATIC RECOGNITION OF PHONEMES USING A SYNTACTIC PROCESSOR FOR ERROR CORRECTION		5. TYPE OF REPORT & PERIOD COVERED MS Thesis
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Robert B. Taylor 2nd Lt.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE December, 1980
		13. NUMBER OF PAGES 86
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Fredric C. Lynch
Approved for public release; IAW AFR 190-17
FREDRIC C. LYNCH, Major, USAF
Director of Public Affairs

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

AUTOMATIC SPEECH RECOGNITION
AUTOMATIC PHONEME RECOGNITION
SYNTACTIC ERROR CORRECTION

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An algorithm was developed and partially implemented to integrate the use of a phoneme recognizer and a syntactic error corrector for continuous speech recognition. The recognizer uses LPC reflection coefficients as a feature set and makes decisions based on the computation of pairwise likelihood ratio tests for M phonemes. The syntactic error corrector uses a backtracking perser to perform phonological rule and grammatical error correction. A computer program is included to provide interactive training with a Tektronix 4010 terminal on a Data General NOVA/ECLIPSE computer system.